البرمجة

1. ادخال البيانات
2. معالجة البيانات
3. استخراج النتائج

To solve a problem $\xrightarrow{YouNeed}$ Program

To create a program $\xrightarrow{YouNeed}$ Algorithm

To create an algorithm $\xrightarrow{YouNeed}$ Tools

To implement tools $\xrightarrow{YouNeed}$ programming Language

Basic Programming Language

Beginners All Purpose Symbolic Instruction Code

Operations that can be done by any computer Language are

1. Input / Output Commands

a. Print

a. A Print "basic language"
b. Print 5
c. Print 5+10
d. Print X
e. Print "result =", 10+5
f. Print x(i)

print X,Y,Z

print "result =";10+5

b. Read, data

```
10 read X,Y
20 data 10,20
```

c. Input

```
Input X
Input a$
Input a(j)
```

Ex:

```
10 input x
20 input y,z
30 input "enter value "; w
40 Read a,b,c
```

```
50 Data 10, 12, 44
60 print "the value of x = ";x
70 print y,z,w
80 end
```

2. Let, assignment =

a) Mathematical Operations

^,*,/,+,-,mod

Mod operator

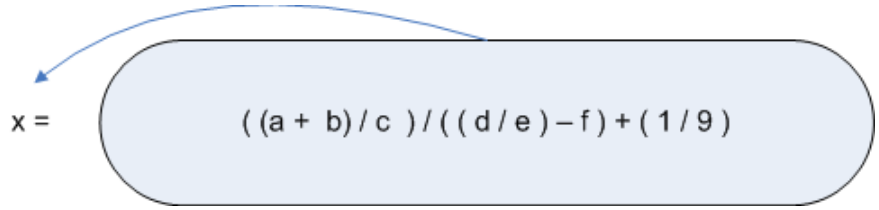
y= 10 mod 3 ans y=1

z= 8 mod 3 ans z=2

Priorities (), ^ , */ mod, + –

$\frac{x * y}{n+3}$ will be written as x*y/(n+3)

$x = \frac{\frac{a+b}{c}}{\frac{d}{e}-f} + \frac{1}{9}$ will be written as (the right hand side will be evaluated first then the value will be given to the left hand side variable)



b) Logical Operations or Relational Operations

Relational operators

Operator	meaning	المعنى
>	Greater than	اكبر من
>=	Greater or equal	اكبر او يساوي
<	less than	اصغر من
<=	less or equal	اصغر او يساوي

=	equal	يساوي
<>	not equal	لا يساوي

(condition) gives either True or False

AND, OR, NOT

1. AND Truth Table

A	B	A AND B
T	T	T
T	F	F
F	T	F
F	F	F

2. OR Truth Table

A	B	A OR B
T	T	T
T	F	T
F	T	T
F	F	F

2. NOT Truth Table

A	NOT A
T	F
F	T

3. REM

```
10 REM this program is to print the area of a square
20 input x
30 print x*x
40 end
```

4. Branching

a) Unconditional branching

10

20
30
40 goto 10

10
20 goto 40
30
40

b) Conditional branching
On x goto 10, 20, 30, …

5. Comparison (IF Statement)

يستعمل امر if لغرض المقارنة عن طريق فحص الحالة condition والتصرف حسب نتيجة المقارنة فاذا كانت الحالة true فيتم تنفيذ الجملة التي تبع كلمة then واذا كانت false فسوف لن يتم تنفيذ الجملة التي بعد كلمة then وفي كلا الحالتين يتم تنفيذ الجملة التي تلي جملة if (الا اذا كانت الجملة التي تلي then هي go to وكانت الحالة true)

It has 5 types
a) **If** ***(condition)*** **then** ***statement***
b) IF ... ENDIF block
IF (CONDITION) THEN
STAT. NO. يتم تنفيذ هذه الجمل
STAT. NO. عندما تكون الحالة صحيحة
.
.
.
ENDIF
c) IF .. Else... ENDIF block
IF (CONDITION) THEN
STAT. NO. يتم تنفيذ هذه الجمل
STAT. NO. عندما تكون الحالة صحيحة
.
.
ELSE
STAT. NO. يتم تنفيذ هذه الجمل
STAT. NO. خطأ عندما تكون الحالة
.
.
ENDIF

d) IF .. Elseif … Elseif... ENDIF block

IF (CONDITION 1) THEN
STAT. NO. يتم تنفيذ هذه الجمل
STAT. NO. عندما تكون الحالة صحيحة
.
.
ELSEif (CONDITION 2) then
STAT. NO. يتم تنفيذ هذه الجمل
STAT. NO. خطأ عندما تكون الحالة
.
.
Elseif (CONDITION 3) then
STAT. NO. يتم تنفيذ هذه الجمل
STAT. NO. عندما تكون الحالة خطا
.
.
Else
STAT. NO. يتم تنفيذ هذه الجمل
STAT. NO. عندما تكون الحالة خطا

ENDIF

e) Nested if

IF .. Else if … Else if... ENDIF block
IF (CONDITION 1) THEN
STAT. NO. يتم تنفيذ هذه الجمل
STAT. NO. عندما تكون الحالة صحيحة
.
.
ELSE if (CONDITION 2) then
STAT. NO. يتم تنفيذ هذه الجمل
STAT. NO. خطأ عندما تكون الحالة
.
.
Else if (CONDITION 3) then
STAT. NO. يتم تنفيذ هذه الجمل
STAT. NO. عندما تكون الحالة خطا
.
.
Else
STAT. NO. يتم تنفيذ هذه الجمل
STAT. NO. عندما تكون الحالة خطا

ENDIF
ENDIF
ENDIF

6. Loops

a) FOR *variable* = *initial value* TO *end value* [STEP *increment* | *decrement*]

يستخدم هذا الامر لغرض تكرار الاوامر المحصورة بين بداية حلقة التكرار المتمثلة بـامر FOR ونهاية حلقة التكرار المتمثلة بامر NEXT . تتكون هذه الجملية من كلمة FOR بعدها اسم متغير مثل I, J, K او اسم تختاره ، value initial تمثل القيمة الابتدائية او الاولية لهذا المتغير، اما end value فتمثل القيمة النهائية للمتغير ، وفي حالة الرغبة بالزيادة بمقدار +1 فلا تحتاج الى ذكر الامر STEP وبعكسه فنحتاج الى ذكر مقدار الزيادة (increment او النقصان decrement) بعد كلمة STEP .

ونستطيع معرفة عدد مرات التكرار من خلال القانون التالي :

$$\text{Number of Loops} = \frac{\text{end value} - \text{initial value}}{Step} + 1$$

امثلة على استخدام الـ FOR

- FOR I = 1 TO 10
- FOR J = 2 TO 20 STEP 2
- FOR K = 1 TO 5 STEP 0.5
- FOR Z= 10 TO 1 STEP -1

...

NEXT *variable*

والخاطئة لحلقات التكرار بعض الاستعمالات الصحيحة

```
FOR I = .....
.........
.........
NEXT I

FOR J=.........
.........
.........
.........
NEXT J
```

```
FOR I = .....
.........
.........
FOR J=....

NEXT I
.........
.........
.........
NEXT J
```

مثال طباعة جملة Basic Language باستعمال حلقة التكرار FOR... NEXT

EXAMPLE
10 INPUT N
20 FOR I = 1 TO N
30 PRINT "BASIC LANGUAGE"
40 NEXT I
50 END

b) do While (condition) … Loop
CLS
c = 0
DO WHILE c < 5
PRINT "basic"
c = c + 1
LOOP
PRINT

c) Do … loop while (condition)
c = 0
DO
PRINT "basic"
c = c + 1
LOOP WHILE c < 5
PRINT

d) do until (condition) … loop

c = 0
DO UNTIL c = 5
PRINT "basic"
c = c + 1
LOOP
PRINT

e) do… loop until (condition)

```
c = 0
DO
PRINT "basic"
c = c + 1
LOOP UNTIL c = 5
```

f) While ...Wend Loop

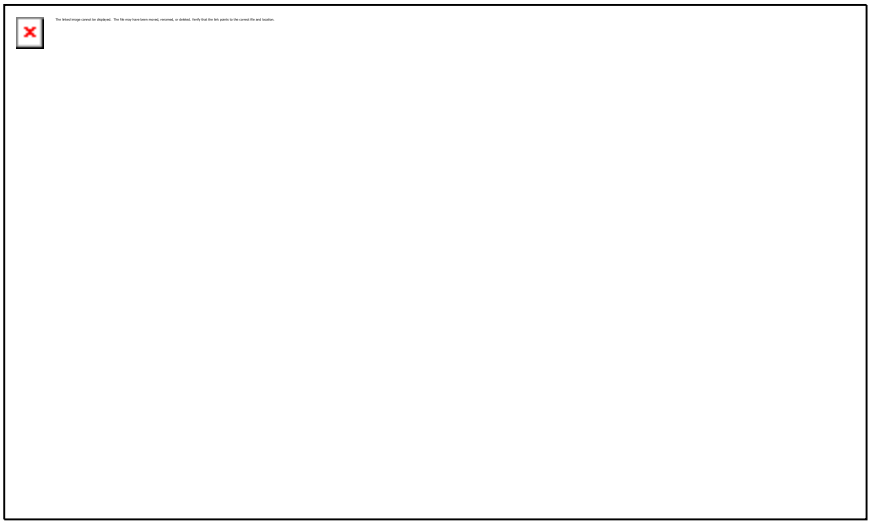
```
WHILE (CONDITION)
...
...
...
WEND
```

هذا نوع اخر من انواع حلقات التكرار حيث يعتمد على الحالة Conditionاذا كانت صحيحة TRUE فسوف يتم تنفيذ الخطوات داخل حلقة التكرار وبعكسه اذا كانت الحالة خطأ FALSE فسوف يتم الخروج من حلقة التكرار الى الجملة التي تلي عبارة WEND

Ex: Print "Basic Programming" 5 times

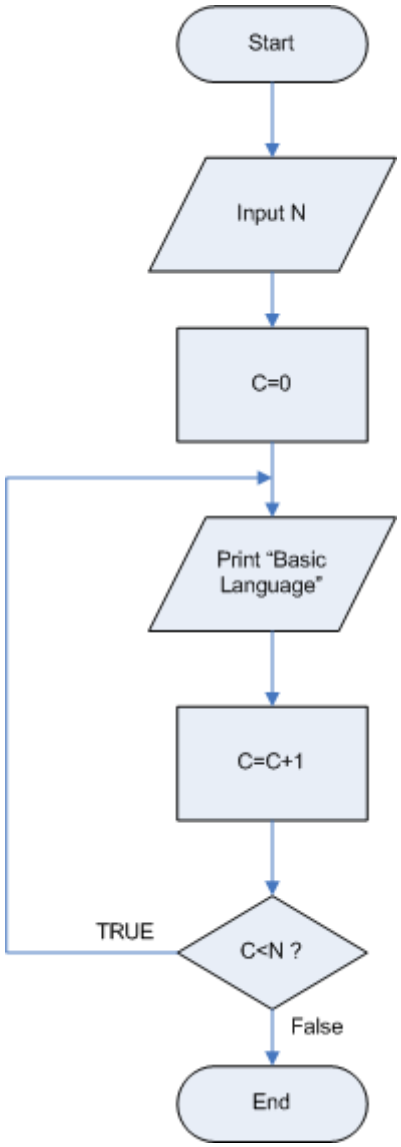
```
10 c=0
20 print "Basic Programming"
30 c = c + 1
40 if c < 5 then go to 20
50 End
```

Flowcharting



Ex:

This is a flow chart for the example shown previously that print the sentence "Basic Language" N times



10- Built in functions الدوال المبنية OR Internal Functions الدوال الداخلية

1- INT( )

هذه الدالة تعيد الجزء الصحيح فقط من الرقم او القيمة المعطاة بين القوسين

example

```
10 y=2.34
20 x=INT(y)
30 print "x=";x
40 end
```

o/p

x=2

2- SQR ( )

هذه الدالة تعيد الجذر التربيعي للقيمة المعطاة بين القوسين

example
```
10 y = 3
20 x= SQR(y)
30 print "x=";x
40 end
```
o/p
x=1.732

3- SIN( )

هذه الدالة تعيد جيب الزاوية المعطاة بين قوسين شرط كون قيمة الزاوية بهيئة زاوية نصف قطرية

example
```
10 x = 30
20 y = x / 180 * 3.141
30 z = sin (y)
40 print z
50 end
```

4- COS()

هذه الدالة تعيد جيب تمام الزاوية شرط كون قيمة الزاوية بهيئة زاوية نصف قطرية

5- ATN()

هذه الدالة تعيد قيمة الزاوية لقيمة الظل المعطاة بين القوسين وتكون قيمة الزاوية المعادة على هيئة زاوية نصف قطرية

TAN = SIN / COS
ARCSIN =ATN(X/SQR(-X*X+1))
ARCCOS= –ATN(-X*X+1)+1.5708

6- ABS( )

هذه الدالة تعيد القيمة المطلقة للقيمة المعطاة بين القوسين

example
```
10 x=5
20 y = - 6
30 print ABS(x), ABS(y)
40 end
```
o/p
5 6

7- LOG( )

هذه الدالة تعيد اللوغارتم الطبيعي للقيمة المعطاة بين القوسين(Ln (Natural Logaritm

8- VAL( )
هذه الدالة تعيد القيمة العددية لعدد تم ادخاله مسبقا على شكل سلسلة حروف String ونحتاج ان نتعامل معه بقيمته العددية

```
10 a$="1234"
20 b$="3120"
30 c=val(a$)
40 d=val(b$)
50 print a$+b$
60 print c+d
end
```

o/p

12343120

4354

9- ASC()
هذه الدالة تعيد رقم الحرف المعطى بين القوسين حسب ترتيبه في الجدول القياسي ASCII table

A 65

B 66

C 67

Z 90

a 97

b 98

c 99

z 122

0 48

1 49

space 32

10 - CHR$()
هذه الدالة تعيد الحرف المعطاة قيمته بين القوسين حسب ترتيبه في الجدول القياسي ASCII table

Random numbers

REQUIRED NUMBER=INT((UPPERBOUND - LOWER BOUND + 1) * RND) + LOWER BOUND

11- Strings

سلاسل الحروف يمكن تعريفها على شكل متغيرات باضافة علامة الدولار S مثل $a

like

10 a$="Basic Language"

هذه الجملة تخزن العبارة المذكورة في متغير في الذاكرة باسم $a

some functions that work with strings

1. LEFT$(STRING,NO.)
2. RIGHT$(STRING,NO.)
3. MID$(STRING,START,NO.)
4. CHR$(NO.)
5. LEN(STRING)
6. ASC(CHARACTER)
7. VAL(STRING)

ex:

```
10 CLS
20 a$=”hello”
30 b$=”world”
40 c$=a$+b$
50 print c$, len(c$)
60 end
```

o/p

helloworld 10

```
10 CLS
20 a$="Basic Language"
30 b$=LEFT$(a$,3)
40 print b$
50 c$=RIGHT$(a$,4)
60 print c$
70 d$=MID$(a$,4,5)
80 print d$
90 e=LEN(a$)
100 print e
110 end
```

o/p

Bas

uage

ic La

14

ex:

```
10 CLS
20 a$ = "1234"
30 b$ = "5678"
40 c$ = a$ + b$
```

```
50 d$ = b$ + a$
60 print c$
70 print d$
80 e = 1234
90 f = 5678
100 g = e + f
110 print g
120 h = VAL(a$)
130 i = VAL(b$)
140 j = h + i
150 print j
160 end
```

o/p

```
12345678
56781234
6912
6912
```

ex: برنامج تحويل الحروف الكبيرة الى حروف صغيرة

```
10 CLS
20 a$="BASIC"
30 L = LEN(a$)
40 for i = 1 to L
50 b$ = MID$(a$,i,1)
60 c = ASC(b$)
70 c = c + 32
80 print CHR$(c)
90 Next i
100 End
```

Review of important algorithms

max. min. number, input data methods

Find the maximum number use terminate condition 4,3,5,6,2,7,–1

```
10 input x
20 max = x
30 input x
40 If x = –1 the goto 70
50 If x > max then max = x
60 go to 30
70 print max
80 End
```

Some Algorithms

Example
Find average of defined numbers (use counter)

```
10 input N
20 S = 0
30 C = 0
40 input X
50 S = S + x
60 C = C + 1
70 if c < N then go to 40
80 A=S/N
90 print "Average=" ; A
100 End
```

Example
Find the maximum number use terminate condition 4,3,5,6,2,7,–1

```
10 input x
20 max = x
30 input x
40 If x = –1 the goto 70
50 If x > max then max = x
60 go to 30
70 print max
80 End
```

نفس البرنامج يمكنه حساب اقل عدد من ضمن مجموعة اعداد بواسطة عمل التحوير التالي

```
10 input x
20 min = x
30 input x
40 If x = –1 the goto 70
50 If x < min then min = x
60 go to 30
70 print min
80 End
```

Example
Find the roots of a second degree equation

```
10 read a,b,c
20 data 4,5,1
30 x=b*b+4*a*c
40 if x > 0 then got to 110
50 If x = 0 then go to 160
```

```
60 print "complex roots"
70 y=SQR(–x)
80 print "first root =";–b/(2*a);"+ i";y/(2*a)
90 print "second root =";–b/(2*a);"– i";y/(2*a)
100 go to 180
110 print "Real different Roots"
120 y= x ^ 0.5
130 print "first root =";(–b + y)/(2*a)
140 print "second root =";(–b – y)/(2*a)
150 go to 180
160 print "equal roots"
170 print "Root equal ="; –b/(2*a)
180 End
```

مثال لطباعة مفكوك عدد باستعمال FOR...NEXT
ملاحظة : مفكوك 5 مثلا هو 5×4×3×2×1 ويرمز له 5!

EXAMPLE
Find the Factorial of a number

```
10 INPUT N
20 F = 1
30 FOR I = 1 TO N
40 F = F * I
50 NEXT I
60 PRINT F
70 END
```

هذا البرنامج يصلح فقط للاعداد التي هي صحيحة واكبر من صفر، والصيغة التالية هي لمعالجة الحالات الاخرى بعد تطوير البرنامج قليلا

```
10 INPUT N
20 IF N < 0 THEN GO TO 120
30 IF N = 0 THEN GO TO 100
40 F = 1
50 FOR I = 1 TO N
60 F = F * I
70 NEXT I
80 PRINT F
90 GO TO 130
100 PRINT 1
110 GO TO 130
120 PRINT "THE NUMBER IS NEGATIVE"
130 END
```

Example
Find Maximum Number Using For...Next loop

```
10 INPUT N
20 INPUT X
30 MAX = X
40 FOR I = 1 TO N-1
50 INPUT X
60 IF X > MAX THEN MAX = X
70 NEXT I
80 PRINT MAX
90 END
```

Example
Print the following Pattern

```
*
**
***
****
*****
```

```
10 FOR I = 1 TO 5
20 FOR J = 1 TO I
30 PRINT  "*" ;
40 NEXT J
50 PRINT
60 NEXT I
70 END
```

Example
Write a computer program to calculate the y value for N terms where:

$$y = 1 + 2x + 3x^2 + 4x^3 + \cdots$$

Note: x is a value given by the user

```
10 INPUT N
20 INPUT X
30 Y=0
40 FOR I = 1 TO N
50 Y=Y + I * X ^ ( I - 1 )
60 NEXT I
70 PRINT "Y=";Y
80 END
```

Example
Write a computer program to calculate the z value for N terms where:

$$z = 1 + \frac{x+1}{2!} + \frac{x+2}{3!} + \frac{x+3}{4!} + \cdots$$

Note: x is a value given by the user

```
10 INPUT N
20 INPUT X
30 Z=1
40 FOR I =2 TO N
50 A=X+I–1
60 B=1
70 FOR J=1 TO I
80 B=B*J
90 NEXT J
100 Z=Z+A/B
110 NEXT I
120 PRINT Z
130 END
```

Example
Write a computer program to calculate the X value for 10 terms where:

$$X = \frac{1}{2} - \frac{4}{5} + \frac{9}{8} - \frac{16}{11} + \cdots$$

```
10 X=0
20 A=1: B=2
30 S = 1
40 FOR I = 1 TO 10
50 X=X+S*A^2/B
60 A=A+1
70 B=B+3
80 S = –S
90 NEXT I
100 PRINT "X=";X
110 END
```

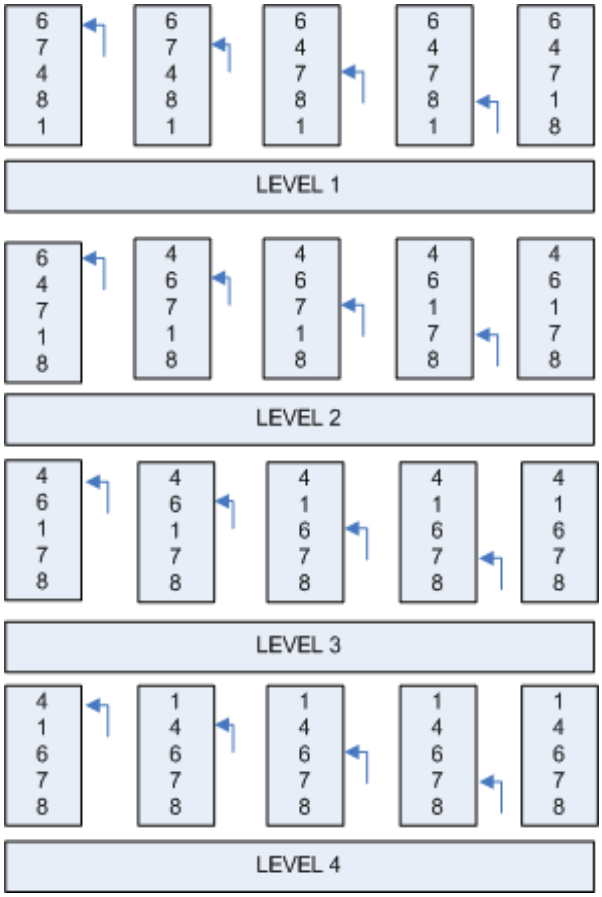
Sorting and Searching الترتيب والبحث

A- SORTING الترتيب

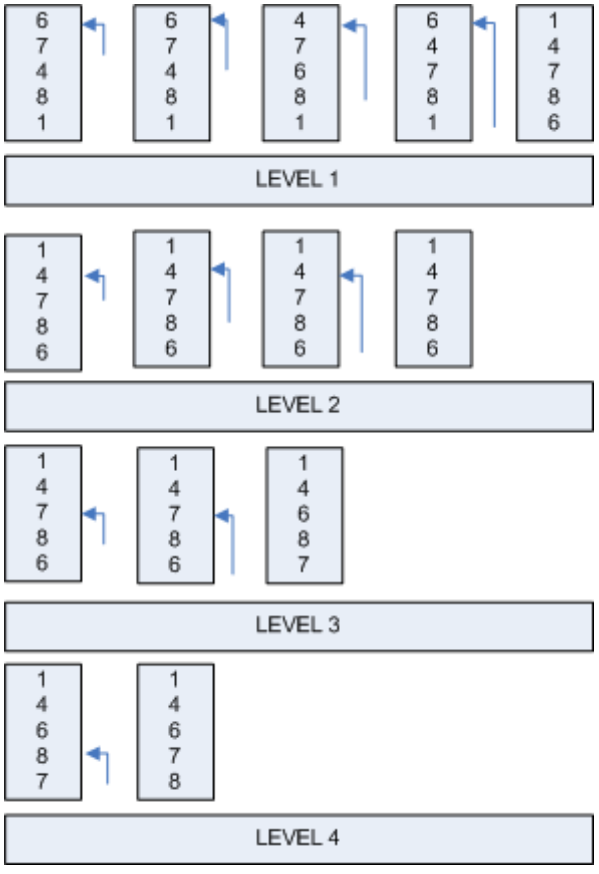
توجد عدة تطبيقات نحتاج فيها الى ان تكون المصفوفة التي نتعامل معها مرتبة اما تصاعديا (ascending) اي ان تكون اقل قيمة في البداية واعلى قيمة في النهاية . او تنازلية (descending) اي اعلى قيمة في الاعلى واقل قيمة في الاسفل

في هذا المخطط يتم شرح خوارزمية الترتيب الفقاعي باستعمال تقنيتان مختلفتان

التقنية الاولى



التقنية الثانية



مثال حسب الخوارزمية الاولى:Ex

```
10 DIM A(6)
20 FOR I=1 TO 6
30 READ A(I)
40 NEXT I
50 DATA 4,7,2,8,6,4
60 FOR I=1 TO 5
70 FOR J=1 TO 5
80 IF A(J+1)>A(J) THEN GOTO 120
90 X=A(J)
100 A(J)=A(J+1)
110 A(J+1)=X
120 NEXT J
130 NEXT I
140 FOR I=1 TO 6
150 PRINT A(I)
160 NEXT I
170 END
```

EX: الخوارزمية الثانية مثال حسب

```
10 DIM A(6)
20 FOR I=1 TO 6
30 READ A(I)
40 NEXT I
50 DATA 4,7,2,8,6,4
60 FOR I=1 TO 5
70 FOR J=I TO 5
80 IF A(J+1)>A(I) THEN GOTO 120
90 X=A(I)
100 A(I)=A(J+1)
110 A(J+1)=X
120 NEXT J
130 NEXT I
140 FOR I=1 TO 6
150 PRINT A(I)
160 NEXT I
170 END
```

Smart sort

مثال لخوارزمية مطورة

```
10 DIM A(6)
20 FOR I=1 TO 6
```

```
30 READ A(I)
40 NEXT I
50 DATA 4,7,2,8,6,4
60 FOR I=1 TO 5
70 F=0
80 FOR J=1 TO 5
90 IF A(J+1)>A(J) THEN GOTO 140
100 X=A(J)
110 A(J)=A(J+1)
120 A(J+1)=X
130 F=1
140 NEXT J
150 IF F=0 THEN GOTO 170
160 NEXT I
170 FOR I=1 TO 6
180 PRINT A(I)
190 NEXT I
200 END
```

Types of Sorting

a. **Selection Sort,**
b. **Bubble Sort,**
c. **Merge Sort,**
d. **Heap Sort,**
e. **Quick Sort,**
f. **Insertion sort**
g. **Others**

B- Searching

1. Linear Search.
2. Binary Search.

1- Linear Search

EX1:

```
10 CLS
20 DIM X(6)
30 FOR I = 1 TO 6
40 READ X(I)
50 NEXT I
60 DATA 3,7,2,3,4,1
70 INPUT "ENTER NUMBER TO SEARCH FOR" ; A
80 FOR I=1 TO 6
90 IF A=X(I) THEN PRINT "FOUND"
100 NEXT I
110 END
```

EX2:

```
10 CLS
20 DIM X(6)
30 FOR I = 1 TO 6
40 READ X(I)
50 NEXT I
60 DATA 3,7,2,3,4,1
70 INPUT "ENTER NUMBER TO SEARCH FOR" ; A
80 FOR I=1 TO 6
90 IF A=X(I) THEN PRINT "FOUND": GOTO 110
100 NEXT I
110 END
```

EX3:

```
10 CLS
20 DIM X(6)
30 FOR I = 1 TO 6
40 READ X(I)
50 NEXT I
60 DATA 3,7,2,3,4,1
70 INPUT "ENTER NUMBER TO SEARCH FOR" ; A
80 F=0
90 FOR I=1 TO 6
100 IF A=X(I) THEN F=1
110 NEXT I
120 IF F=1 THE PRINT "FOUND"
130 IF F=0 THEN PRINT "NOT FOUND"
140 END
```

EX4:

```
10 CLS
20 DIM X(6)
30 FOR I = 1 TO 6
40 READ X(I)
50 NEXT I
60 DATA 3,7,2,3,4,1
70 INPUT "ENTER NUMBER TO SEARCH FOR" ; A
80 C = 0
90 FOR I=1 TO 6
100 IF A=X(I) THEN C = C + 1
110 NEXT I
120 IF C>0 THE PRINT "FOUND "; C
130 IF C=0 THEN PRINT "NOT FOUND"
140 END
```

2- Binary Search

Searching for X in L.

Better way if L is sorted:

1. Compare X to the middle value (M) in L.
2. if X = M we are done.
3. if X < M we continue searching in 1st half of L only.
4. if X > M we continue searching in 2nd half of L only.

Example 1: L = 1 3 4 6 8 9 11. X = 4.

- X < 6: Repeat with L = 1 3 4.
- X > 3: Repeat with L = 4.
- X = 4: We found X!

Example 2

Let's see an example. Say we're searching for the value 37 in the following array:

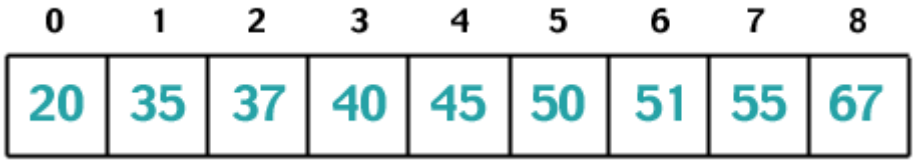


Figure 2.1: Array on which to perform a binary search

We set our low and our high values to be at the beginning and ends of the array, and our middle value to be their average:

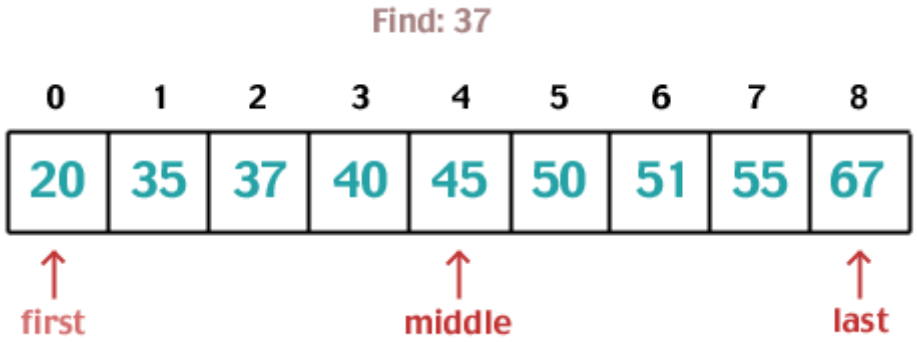


Figure 2.2: Set initial first, middle and last values

We then compare 37 to the value at the middle location. Is 37 = = 45? No, it is less than 45. So we update the last pointer to be middle - 1, and readjust the middle pointer accordingly:

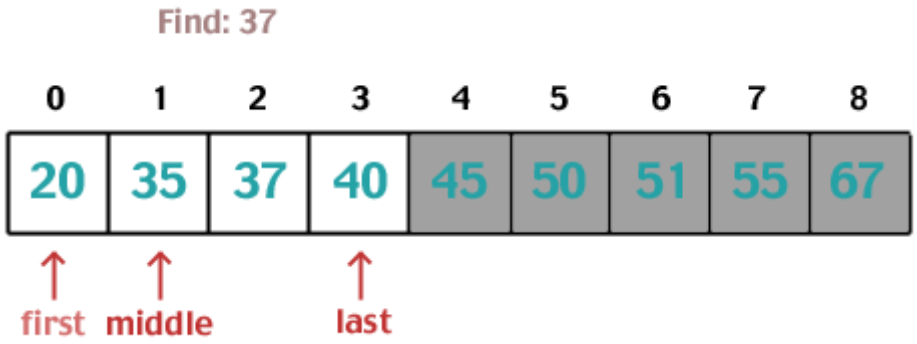


Figure 2.3: Now searching the bottom half

Is 37 = = 35? No. It is greater than 35. So we update the first and middle pointers accordingly:

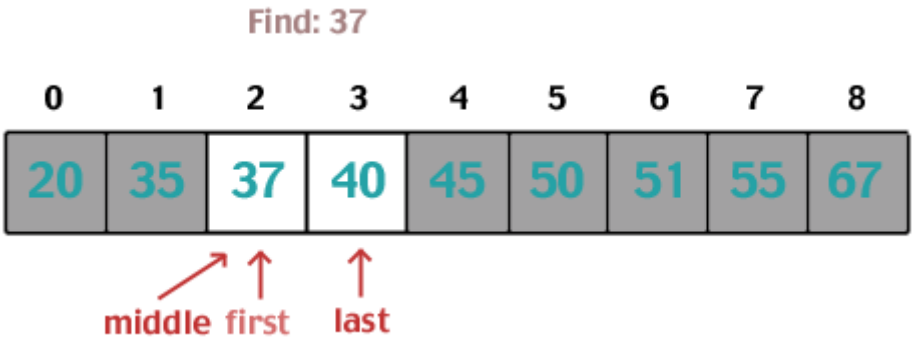


Figure 2.4: Now searching upper half of bottom half

Is `37 == 37`? Yes! We found it:

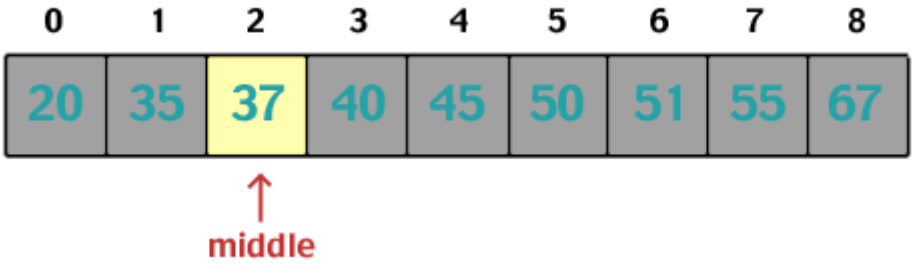


Figure 2.5: We found it!

Matrix transpose – non square matrix

```
10 dim a(2,3) , b(3,2)
20 for I = 1 to 2
30 for j=1 to 3
40 input a(i,j)
50 next j
60 Next i
70 For I = 1 to 2
80 For j = 1 to 3
90 B(j,i)=a(I,j)
100 Next j
110 Next i
120 For I = 1 to 3
130 For j=1 to 2
140 Print b(I,j),
150 Next j
160 Print
170 Next i
180 End
```

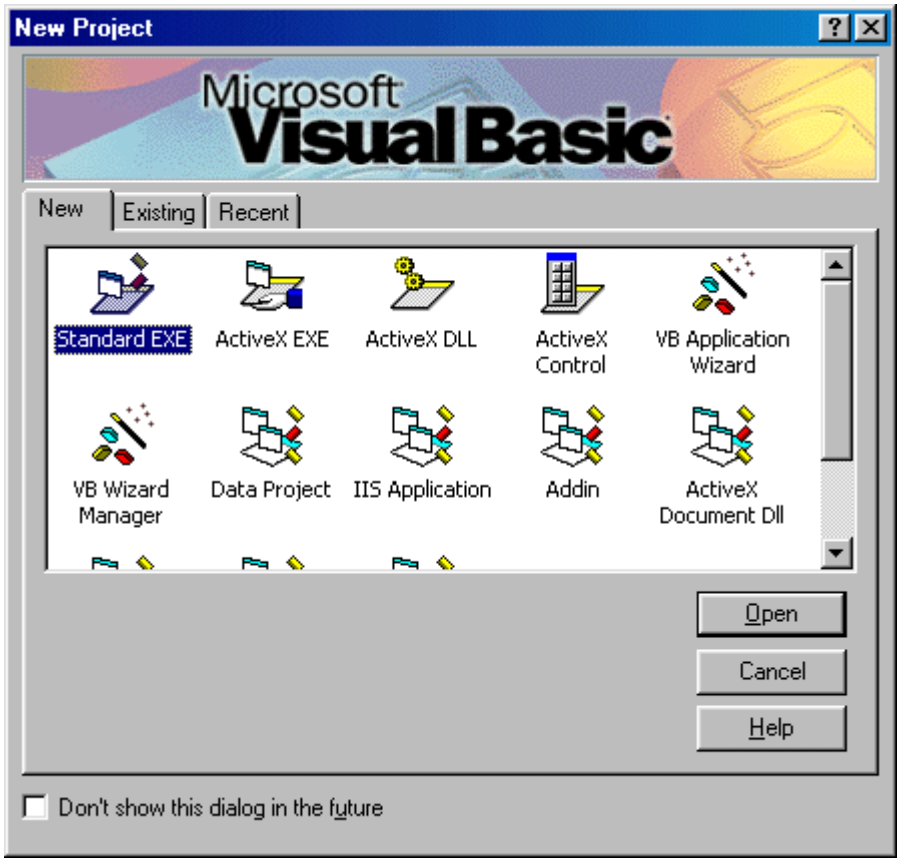
Matrix transpose – square matrix

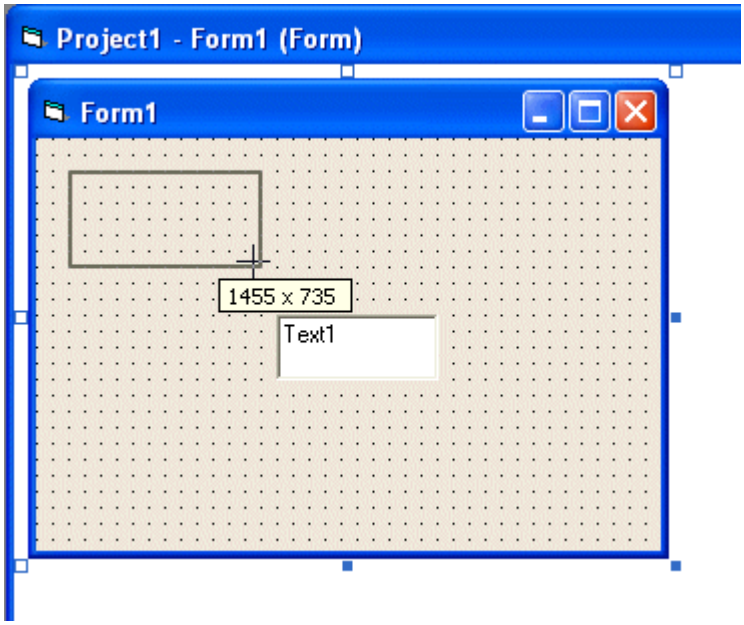
```
10 dim a(3,3)
20 for I = 1 to 3
30 for j=1 to 3
```

```
40 input a(i,j)
50 next j
60 Next i
70 For I = 1 to 3
80 For j = 1 to 3
90 if i>=j then goto  130
100 T=a(I,j)
110 A(I,j)=a(j,i)
120 A(j,i)=t
130 Next j
140 Next i
150 For I = 1 to 3
160 For j=1 to 3
170 Print a(I,j),
180 Next j
190 Print
200 Next i
210 End
```

Visual Basic

1– Visual Basic Environment





Toolbox and Controls

Introduction

A Windows control, also called a control, is an object that allows the user to interact with the computer. Such an object must be displayed on the screen or somehow made available to the user who can then click it, move it, resize it, type in it or retrieve something from it. Because there are so many operations a user can perform on the computer, controls are separate in categories according to their functionality and their roles in an application. Nevertheless, to make your application effective, as the developer, you will decide what the user can do with your application and what should be excluded.

When creating your application, you add controls to it as you judge them relevant for the possible assignments that can be performed on your application. While working, you will deal with two big categories of controls.

Controls Fundamentals

You as the developer will decide what control should be available in your application, what functionality that control should provide, and what the user can do with it. Some of the functionality is controlled by the operating system because such functionality is part of the computer's behavior. Some other aspects are under your control.

When creating your application, you will most likely start from a form. Other controls are added to the form. To use one of them, you will get it from an object called the Toolbox and then add them to the form. Once a control is available to you, you can customize its appearance and behavior.

To implement their intended assignment, one of the most regular operations a control can perform is to fire events.

Control Selection

To provide the necessary functionality for your application, you will use controls from the Toolbox and add them to another component such as a form. The control you pick up from the Toolbox is also referred to as a

child control. The control or object on which you add a child control is referred to as its parent or host. This can be a form or another object that has the capacity to host other controls.

To identify a control on the Toolbox, you can position the mouse on it. A tool tip would appear.

To add a control to a host, on the toolbox, you can double-click it. Alternatively, you can click the control on the toolbox and then "draw" it on the host. You can keep adding controls to a host as necessary.

If you want to add a control over and over again, you can press and hold Ctrl, click the control on the Toolbox, then draw it in the desired area on the host. Every time you draw, the control would be added to the form or host. Once you have added enough controls, you can release Ctrl.

If you select a control by mistake, you can simply click another. The new one would become selected. If you clicked a control but don't want any control at all, you can click the Pointer button. You cannot select more than one control to add to a host.

Properties of Controls

Introduction

If you access a code when designing the application, it is said that you are working at design time. If you access a control with code, it is said that you are at run time. Therefore, design time refers to the form being designed while displaying in Visual Basic. Run time refers to the time the control is displaying to the user.

After adding a control to the application, you can customize it. For example, you can change some parts of its appearance. You can also give it assignments. These are done from two parts: the Properties window and the Code editor.

Controls are broadly classified in two groups. A control is referred to as graphical if the user can see it. There are other controls that will work behind the scenes at run time. Such control are not graphical (an example is the Timer). They can be referred to as static. The user never sees these controls. There are some other controls not considered graphical because the user cannot directly change their values. For example, a control that displays only text (such is the case for the Label) is not considered graphical.

A Windows control is an object that imitates a real world object. As such, it is made of characteristics that define it. A characteristic is also called a property. A property is any aspect that describes an object.

Once you have a control, you can change its properties in the Properties window. This is considered that you are controlling the properties at "design time". To change the properties of a control, first select it, then proceed with changing the desired properties in the Properties window.

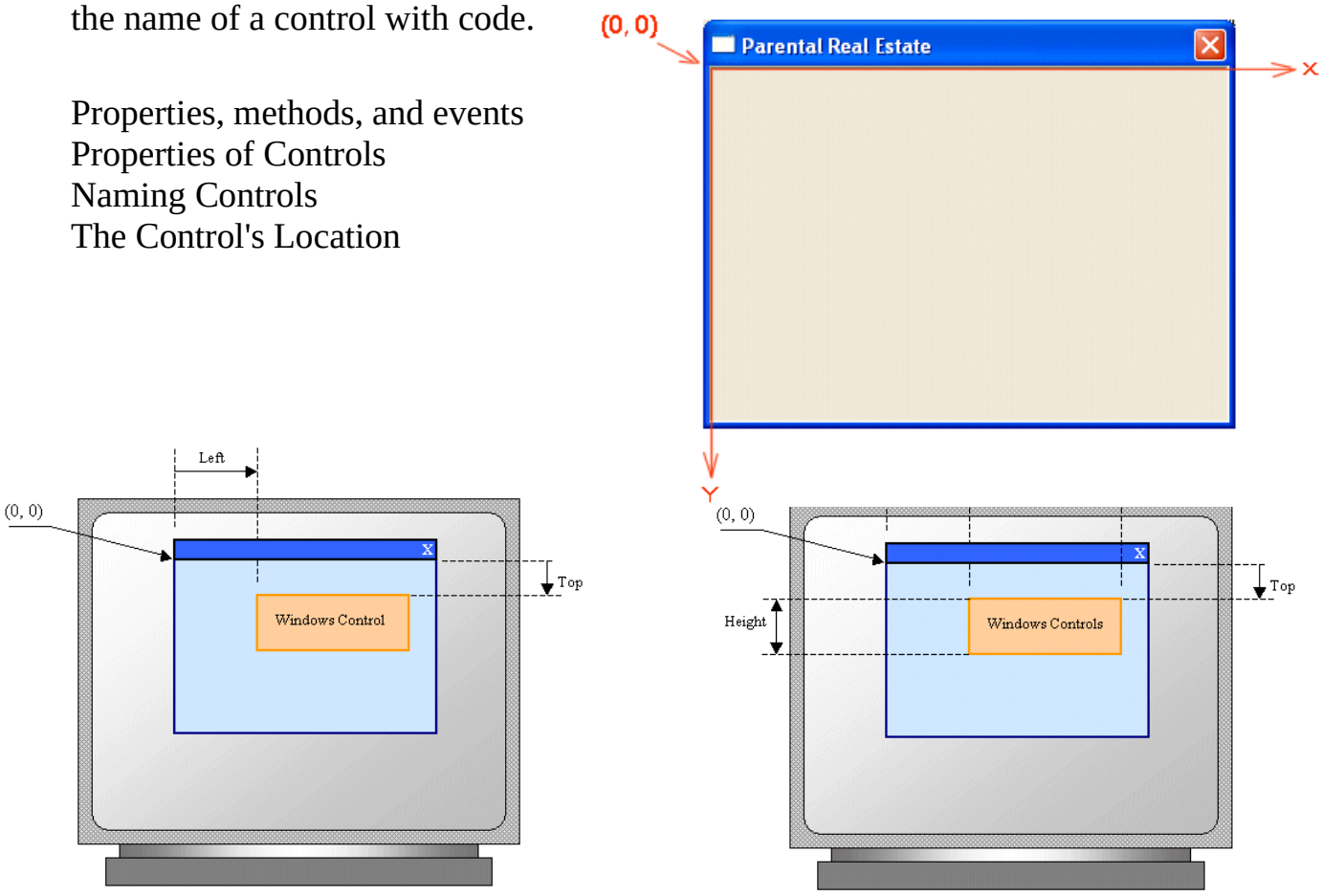
To control a form's properties with code, you will refer to itself. A form refers to itself using the **Me** keyword. To change the properties of a control with code, you refer to it by its name. Whether dealing with a form or a control, after typing **Me** for the form, or the name of the control, type a period. A list of the properties (and possibly other objects that we will know eventually) will appear. You can continue typing or simply select from the list. And continue with your coding. Not all properties can be changed with code.

Controls Names

Everything on a computer must have a name. In the same way, to refer to a control in your code, you must give it a name. When you add a new control to your application, it receives a default name. When necessary, which will be almost all the time, you should change that name to a more recognizable one.

To change the name of a control, first select it. Then, in the Properties window, click (Name) and type the desired name. Refrain from changing the name of a control with code.

Properties, methods, and events
Properties of Controls
Naming Controls
The Control's Location



Controls Text and Caption

Some controls are meant to display or sometimes request text from the user. For such controls, this text is referred to as caption while it is simply called text for some other controls. This property is not available for all controls.

If a control displays text, it then has a **Caption** or a **Text** field in the Properties window. After adding such a control to a form, its **Caption** or its **Text** field may the same text as its name. At design time, to change the text of the control, click either its **Caption** or its **Text** field and type the desired value. For most controls, there are no strict rules to follow for this text. Therefore, it is your responsibility to type the right value.

The text provided in a **Caption** or **Text** fields of a text-based control can only be set "as is" at during design. If you want the text to change while the application is running, you can format it. For example, such a control can display the current time or the name of the user who is logged in. These format attributes cannot be set at deign time. To change the text of a text-based control at run time, either assign a simple string or provide a formatted string to either the **Caption** or the Text property.

Control's Visibility

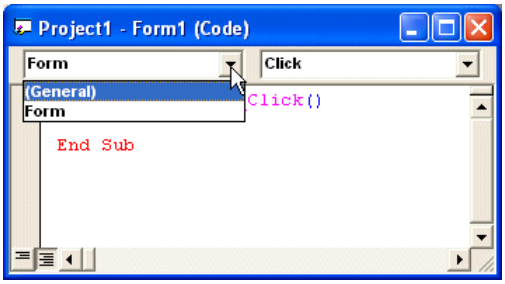
Visible property set to True. hide a control, set its Visible property to False

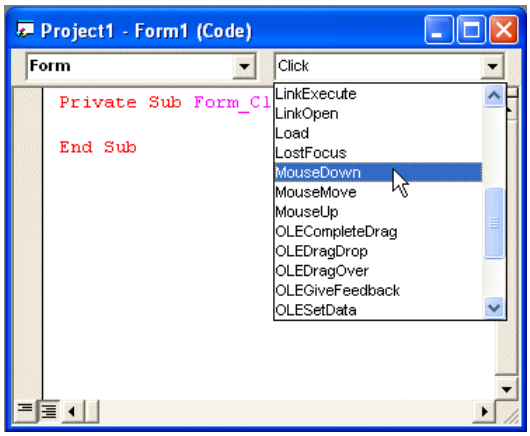
Control's Availability

When a control is enabled, the user can click it or type in it. You can also prevent this type of action by disabling the control

Tab Sequence

This means that, whether you click an object, type something using the keyboard, move the mouse on the screen, or press and click at the same time, a message is composed





Categories of Events
The Keyboard Events
The Click Event
The Double-Click Event
The Right-Click Event
The Focus Events

Introduction to Data Types
String
Dim VariableName As DataType
Dim CountryName As String

CountryName = ""

Boolean
Dim IsMarried As Boolean

IsMarried = False

Byte
Dim StudentAge As Byte

Integer
Dim MusicTracks As Integer

Decimal Data Types
A real number is one that displays a decimal part. This means that the number can be made of two sections separated by a symbol that is referred to as the Decimal Separator or Decimal Symbol. This symbol is different by language, country, group of languages, or group of countries.

Single
A single is a decimal number whose value can range from –3.402823e38 and –1.401298e-45 if the number is negative, or 1.401298e-45 and 3.402823e38 if the number is positive

Dim Distance As Single

Double
This is used for a variable that would hold numbers that range from 1.79769313486231e308 to –4.94065645841247e–324 if the number is negative or from 1.79769313486231E308 to 4.94065645841247E–324 if the number is positive.

Dim Distance As Double

Currency
Dim StartingSalary As Currency
Date
Dim DateOfBirth As Date

Object
An Object is almost anything else that you want to use in your program.

Variant
A Variant can be used to declare any kind of variable. You can use a variant when you can't make up your mind regarding a variable but, as a beginning programmer, you should avoid it.
Here is a table of various data types and the amount of memory space each one uses:

Data type	Description	Range
Byte	1-byte binary data	0 to 255
Integer	2-byte integer	– 32,768 to 32,767
Long	4-byte integer	– 2,147,483,648 to 2,147,483,647
Single	4-byte floating-point number	– 3.402823E38 to – 1.401298E – 45 (negative values)
		1.401298E – 45 to 3.402823E38 (positive values)
Double	8-byte floating-point number	– 1.79769313486231E308 to – 4.94065645841247E – 324 (negative values)
		4.94065645841247E – 324 to 1.79769313486231E308 (positive values)
Currency	8-byte number with fixed decimal point	– 922,337,203,685,477.5808 to 922,337,203,685,477.5807
String	String of characters	Zero to approximately two billion characters
Variant	Date/time, floating-point number, integer, string, or object. 16 bytes, plus 1 byte for each character if a string	Date values: January 1, 100 to December 31, 9999 Numeric values: same range as Double String values: same range as String Can also contain Error or Null values

	value.	
Boolean	2 bytes	True or False
Date	8-byte date/time value	January 1, 100 to December 31, 9999
Object	4 bytes	Any Object reference

Constants
The Carriage Return-Line Feed Constant
vbCrLf

Operators
^,+,-,*,/,\ integer division,mod,&,_ line continuation
String concatination
dblResult = dblOperand1 + " " + dblOperand2
remarks
' This line will not be considered as part of the code
    Rem I can write anything I want on this line
Logical operators
=,<>,<,>,<=,>=
NOT or and
If statement

If *ConditionToCheck* **Then** *Statement*
If *Condition* **Then**
    Statement1
    Statement2
    Statementn
End If
If *ConditionToCheck* **Then**
    Statement1
Else
    Statement2
End If

If *Condition1* **Then**
    Statement1
ElseIf *Condition2* **Then**
    Statement2
ElseIf *Conditionk* **Then**

Statementk
End If

Select Case *Expression*
Case *Expression1*
Statement1
Case *Expression2*
Statement2
Case *Expressionk*
Statementk
End Select

Loops Repeaters

The Do...While Loop

The formula of the **Do... While** loop is:

Do While *Condition*
Statement(s)
Loop

Do
Statement(s)
Loop While *Condition*

Do Until *Condition*

Statement(s)
Loop

Do
Statement(s)
Loop Until *Condition*

For *Counter* = *Start* **To** *End*
Statement(s)
Next

For *Counter* = *Start* **To** *End* **Step** *Increment*
Statement(s)

Next *Counter*

For Each *Element* **In** *Group*
Statement(s)
Next *Element*

Built-in Logical Constants: NULL

A variable is said to be null when its value is invalid or doesn't bear any significant or recognizable value.

Built-in Logical Constants: TRUE and FALSE

An expression is said to be false if the result of its comparison is 0. Otherwise, the expression is said to bear a true result.

Message Boxes
MsgBox *Message*, [*Buttons*], [*Title*], [*HelpFile*], [*Context*]

Button	Value	Display
vbOKOnly	0	OK
vbOKCancel	1	OK Cancel
vbAbortRetryIgnore	2	Abort Retry Ignore
vbYesNoCancel	3	Yes No Cancel
vbYesNo	4	Yes No
vbRetryCancel	5	Retry Cancel

Icon	Value	Description
vbCritical	16	
vbQuestion	32	
vbExclamation	48	
vbInformation	64	

Here is an example:

```
Private Sub Form_Load()
   MsgBox "Are you ready for exam", vbYesNo Or vbQuestion
End Sub
```

When treated as a function, MsgBox returns a value. This value corresponds to the button the user clicks on the message box. Depending on the buttons the message box is displaying, after the user has clicked, the **MsgBox** function can return one of the following values:

Button	Return	Value
OK	vbOK	1
Cancel	vbCancel	2
Abort	vbAbort	3
Retry	vbRetry	4
Ignore	vbIgnore	5
Yes	vbYes	6
No	vbNo	7

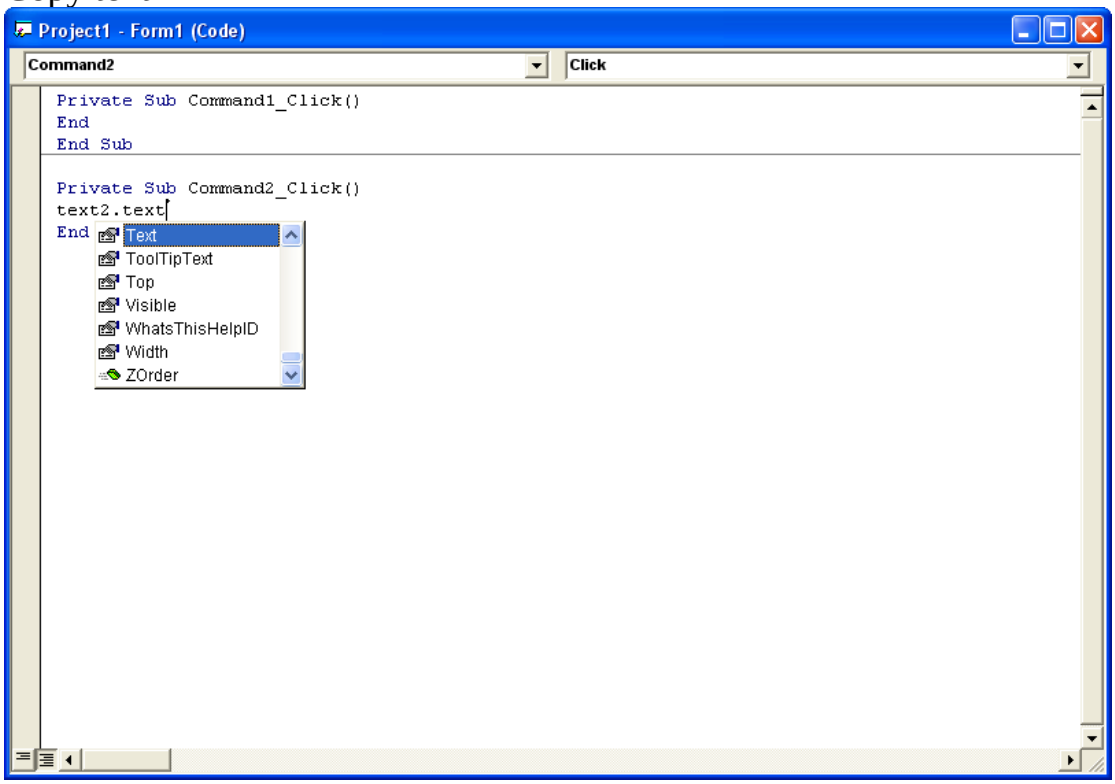
Table 6.3. The default buttons displayed in a message box.

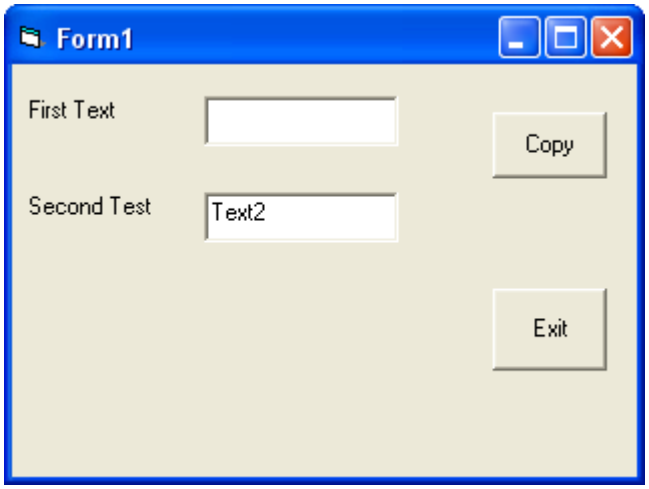
Named Literal	*Value*	*Description*
`vbDefaultButton1`	`0`	The first button is the default.
`vbDefaultButton2`	`256`	The second button is the default.
`vbDefaultButton3`	`512`	The third button is the default.

The following MsgBox() function produces the message box shown in Figure 6.4:

```
intPress = MsgBox("Are you ready for the report?", vbQuestion + _
  vbYesNoCancel, "Report Request")
```

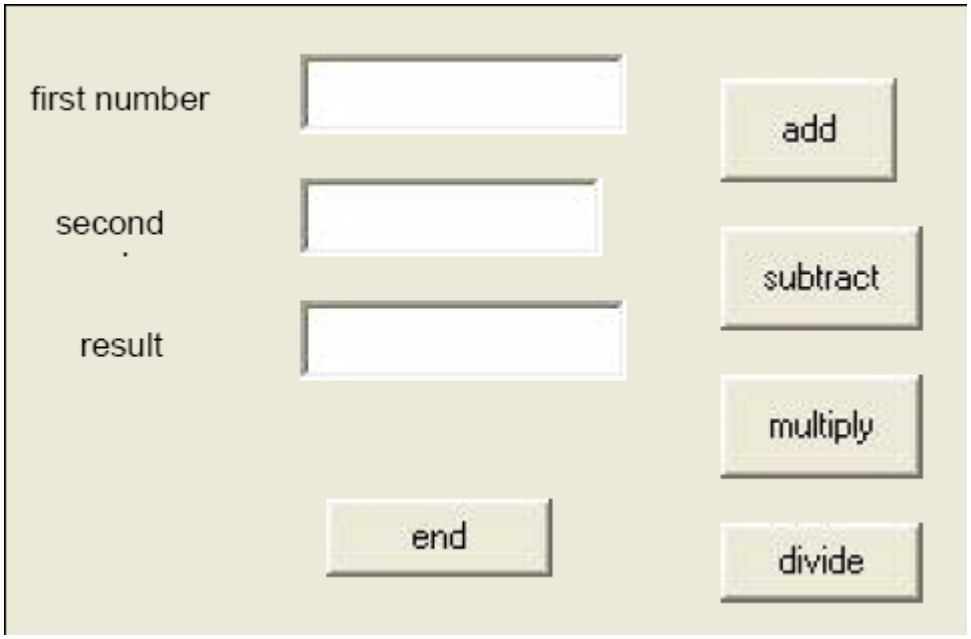
First Program
Copy text






```
Private Sub Command1_Click()
End
End Sub

Private Sub Command2_Click()
Text2.Text = Text1.Text
End Sub
```

TextBox, Command Button, Label

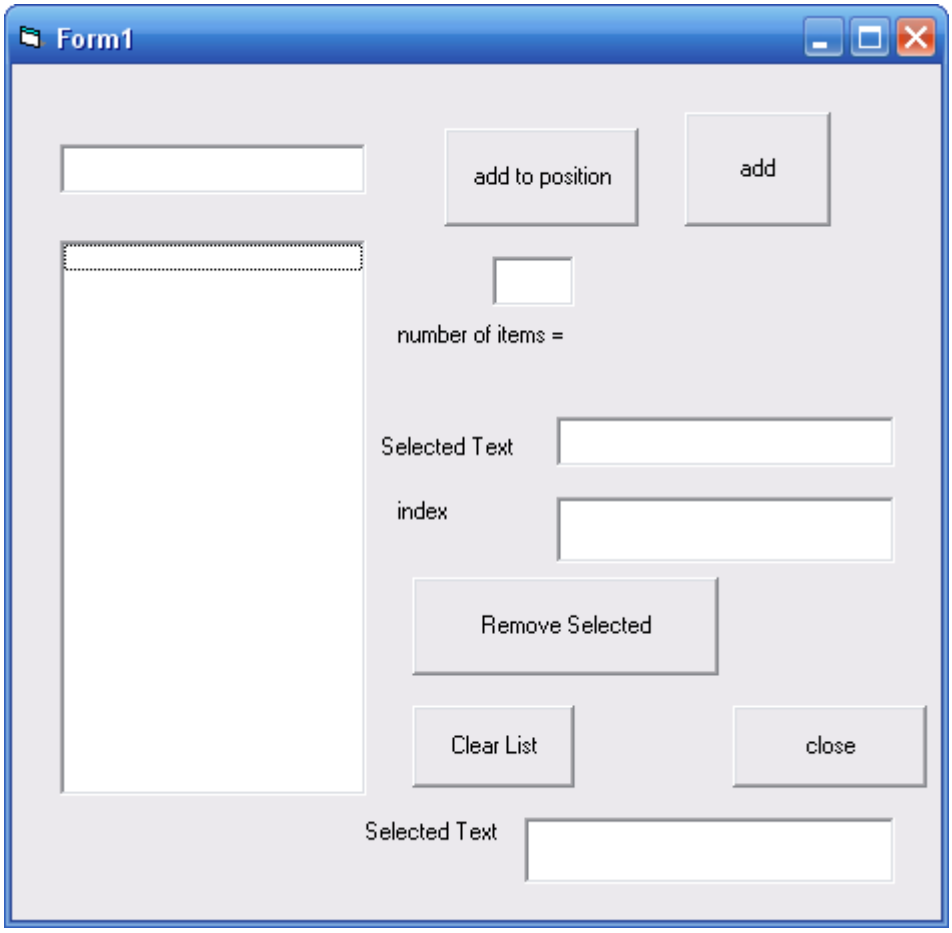



```
Private Sub Command1_Click()
Text3.Text = Val(Text1.Text) + Val(Text2.Text)
End Sub
Private Sub Command2_Click()
Text3.Text = Val(Text1.Text) - Val(Text2.Text)
End Sub
Private Sub Command3_Click()
Dim x As Double
Dim y As Double
Dim z As Double
x = Val(Text1.Text)
y = Val(Text2.Text)
z = x * y
Text3.Text = z
End Sub
Private Sub Command4_Click()
Text3.Text = Val(Text1.Text) / Val(Text2.Text)
End Sub
Private Sub Command5_Click()
End
End Sub

TextBox Text1
Text=""
TextBox Text2
```

Text=""
TextBox Text3
Text=""
CommandButton Command1
Caption = "add"
CommandButton Command2
Caption = "subtract"
CommandButton Command3
Caption = "multiply"
CommandButton Command4
Caption = "divide"
CommandButton Command5
Caption = "end"

Label Label1
Caption = "first number"
Label Label2
Caption = "second number"
Label Label3
Caption = "result"

List Box



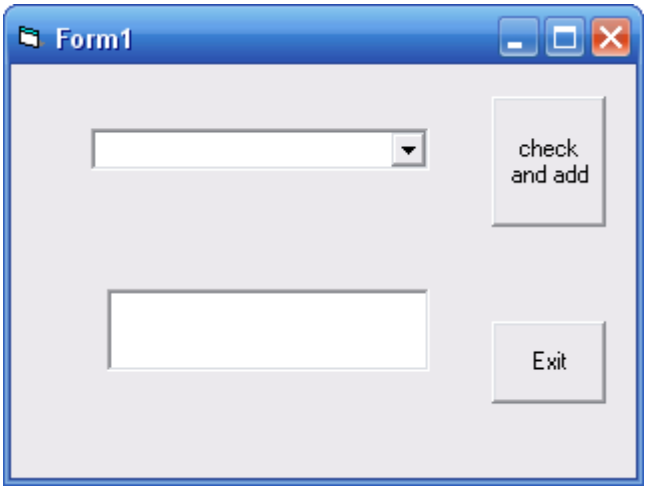

```
Private Sub Command1_Click()
If Trim(Text1.Text) = "" Then
MsgBox "text is empty"
Else
List1.AddItem Text1.Text
Label1.Caption = "number of items = " + Str(List1.ListCount)
End If
End Sub

Private Sub Command2_Click()
End
End Sub

Private Sub Command3_Click()
If Trim(Text1.Text) = "" Or Trim(Text2.Text) = "" Then
MsgBox "empty text"
Else
If Val(Text2.Text) > List1.ListCount Then
```

```
MsgBox "position is greater than the list"
Else
List1.AddItem Text1.Text, Val(Text2.Text)
Label1.Caption = "number of items = " + Str(List1.ListCount)
End If
End If
End Sub

Private Sub Command4_Click()
If List1.ListIndex = -1 Then
MsgBox "Please select an item"
Else
List1.RemoveItem List1.ListIndex
Label1.Caption = "number of items = " + Str(List1.ListCount)
End If
End Sub

Private Sub Command5_Click()
List1.Clear
Label1.Caption = "number of items = " + Str(List1.ListCount)

End Sub

Private Sub List1_Click()
Text3.Text = List1.Text
Text4.Text = List1.ListIndex
Text5.Text = List1.List(List1.ListIndex)
End Sub


List1.columns = 0 | 1…n
List1.sorted = True | False
List1.visible = True | False
List1.Enabled = True | False
```

ComboBox

Styles

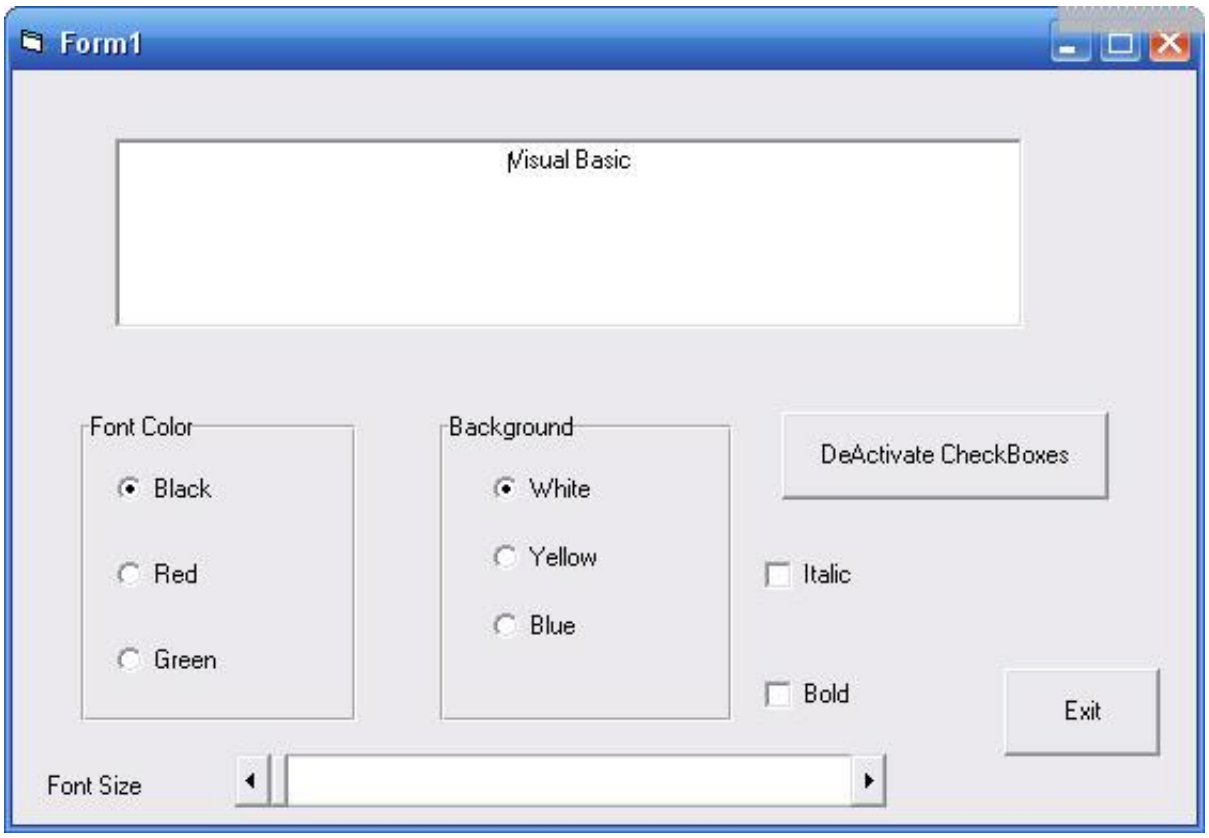
0 Combo Box
1 Simple Combo = textbox
2 Drop Down List




```
Private Sub Combo1_Click()
Text1.Text = "Your Country is" & vbCrLf & Combo1.Text
End Sub

Private Sub Command1_Click()
Dim st As String
Dim i As Integer
Dim found As Boolean
If Trim(Combo1.Text) = "" Then
MsgBox "empty text"
Exit Sub
Else
st = Trim(Combo1.Text)
End If
found = False
'For i = 0 To Combo1.ListCount –1
'If st = Trim(Combo1.List(i)) Then
'found = True
'Exit For
'End If
'Next
If inlist(st) Then found = True
```

```
If found Then
MsgBox "Item is already in the list"
Else
Combo1.AddItem st
MsgBox "new item " & st & " was added"
End If
End Sub

Private Sub Command2_Click()
End
End Sub

Private Sub Form_Load()
Combo1.AddItem "Iraq"
Combo1.AddItem "Syria"
Combo1.AddItem "Lebanon"
Combo1.AddItem "Egypt"
End Sub

Private Function inlist(a As String) As Boolean
Dim i As Integer
inlist = False
For i = 0 To Combo1.ListCount –1
If a = Trim(Combo1.List(i)) Then
inlist = True
Exit For
End If
Next i
End Function
```

CheckBox, Option, Scroll




```
Private Sub Check1_Click()
If Check1.Value = 0 Then
Text1.FontItalic = False
Else
Text1.FontItalic = True
End If
End Sub

Private Sub Check2_Click()
If Check2.Value = 0 Then
Text1.FontBold = False
Else
Text1.FontBold = True
End If
End Sub

Private Sub Command1_Click()
If Check1.Enabled Then
Check1.Enabled = False
Command1.Caption = "Activate CheckBoxes"
Else
```

```
Check1.Enabled = True
Command1.Caption = "DeActivate CheckBoxes"
End If

If Check2.Enabled Then
Check2.Enabled = False
Else
Check2.Enabled = True
End If
End Sub

Private Sub Command2_Click()
End
End Sub

Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
With Text1
If X < .Left Or X > .Left + .Width Or Y < .Top Or Y > .Top + .Height Then
Text1.Visible = True
End If
End With
End Sub

Private Sub HScroll1_Change()
Text1.FontSize = HScroll1.Value
End Sub

Private Sub Option1_Click()
Text1.ForeColor = vbRed
End Sub

Private Sub Option2_Click()
Text1.ForeColor = vbBlack
End Sub

Private Sub Option3_Click()
Text1.ForeColor = vbGreen
End Sub

Private Sub Option4_Click()
Text1.BackColor = vbWhite
End Sub
```

Private Sub Option5_Click()
Text1.BackColor = vbYellow
End Sub

Private Sub Option6_Click()
Text1.BackColor = vbBlue
End Sub

Private Sub Text1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
Text1.Visible = False
End Sub

Constant	Value	Description
vbBlack	&h00	Black
vbRed	&hFF	Red
vbGreen	&hFF00	Green
vbYellow	&hFFFF	Yellow
vbBlue	&hFF0000	Blue
vbMagenta	&hFF00FF	Magenta
vbCyan	&hFFFF00	Cyan
vbWhite	&hFFFFFF	White

Checkbox Values
0 Unchecked
1 Checked
2 Grayed
- Caption

Option
- Caption

Value True | False

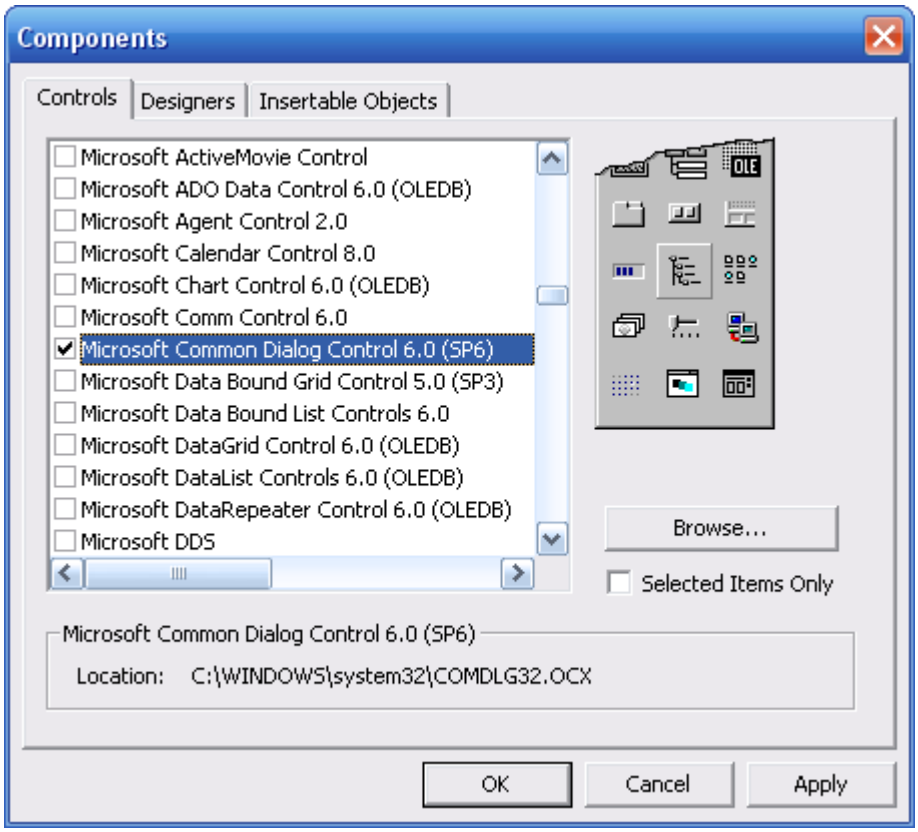
The Input Box

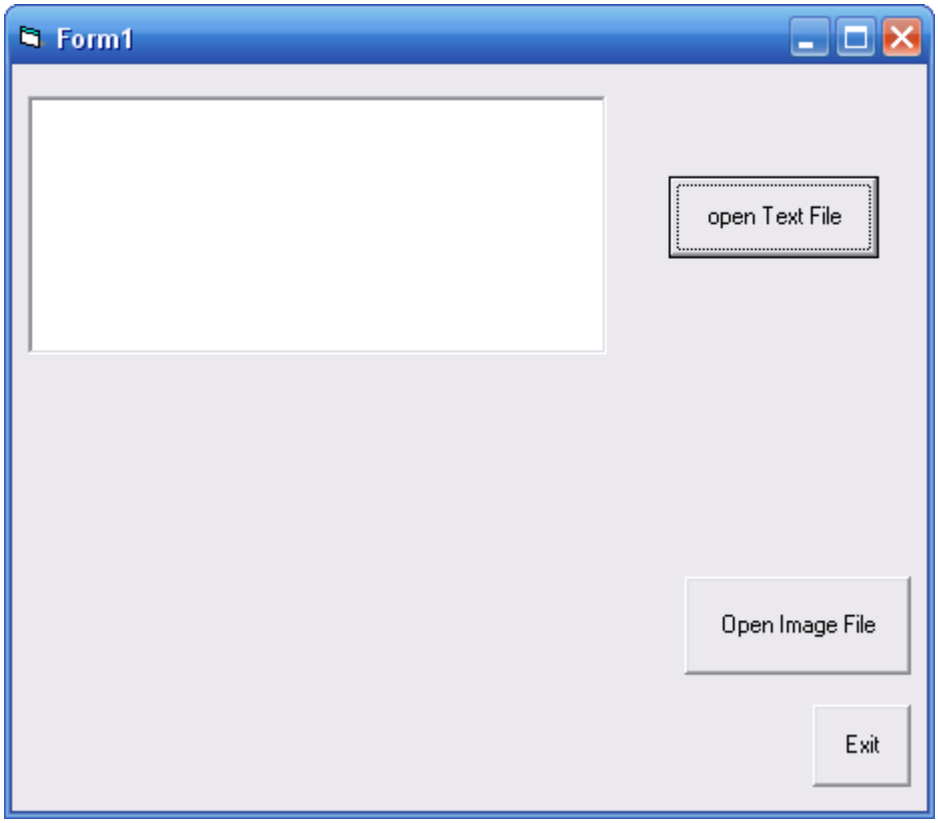
InputBox (prompt[, title][, default][, xpos][, ypos][, helpfile, context])

default String expression displayed in the text box as the default response if no other input is provided. If you omit default, the text box is displayed empty.

The Character To ASCII Conversion
Chr(Number)
Case Conversion
Function UCase(Letter As Char) As Char
Function UCase(Expression As String) As String
Is it Empty?
IsEmpty(Expression)
Is it Null?
IsNull(Expression)

Dialog Box, image

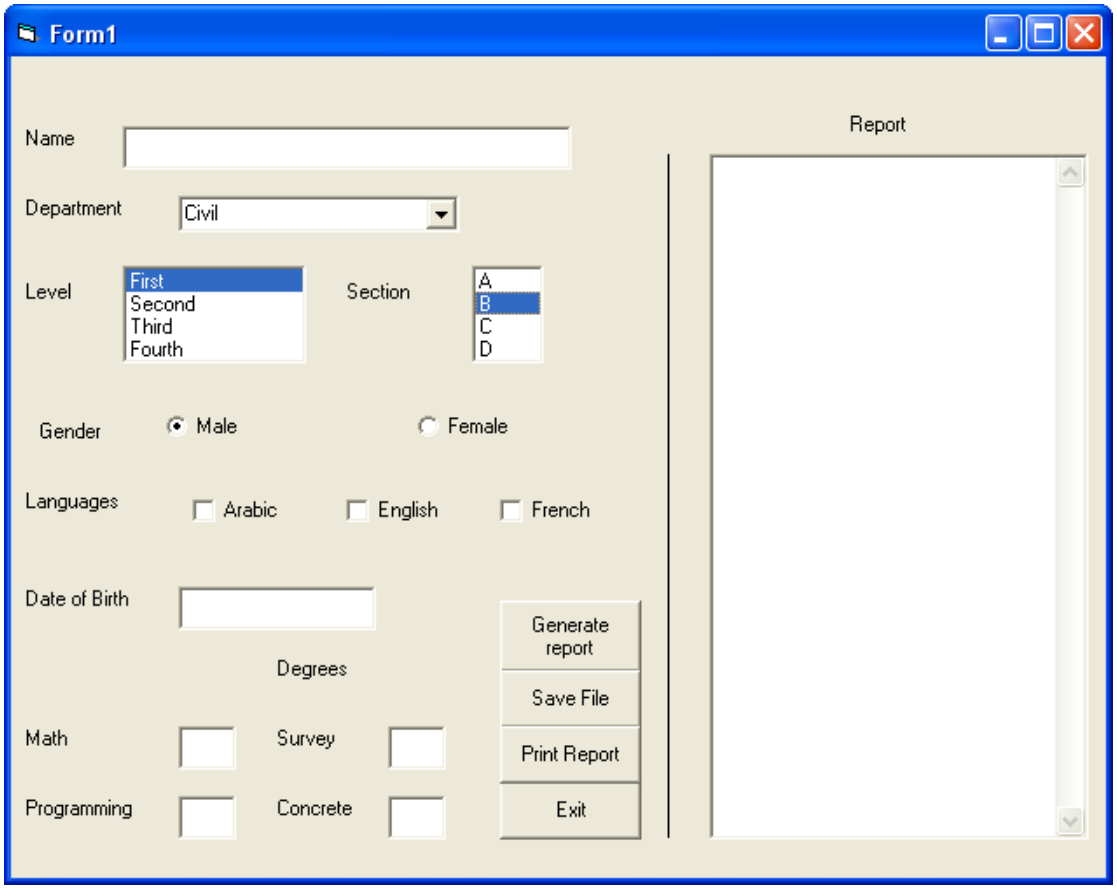





```
Private Sub Command1_Click()

Dim fname As String
Dim filenum As Integer
On Error GoTo last
CD1.Filter = "All Files (*.*)|*.*|Text Files|*.txt"
CD1.FilterIndex = 2
CD1.DefaultExt = "txt"

CD1.CancelError = True
CD1.ShowOpen
If Trim(CD1.FileName) = "" Then
MsgBox "Canceled By the User"
Else
fname = CD1.FileName
End If
' Read the file's contents into the control.
filenum = FreeFile()
Open fname For Input As #filenum
Text1.Text = Input$(LOF(filenum), filenum)
Close #filenum
```

```
last:
End Sub

Private Sub Command2_Click()
End
End Sub

Private Sub Command3_Click()
Dim fname As String
Dim filenum As Integer
On Error GoTo last
CD1.Filter = "Bitmap (*.BMP)|*.BMP|JPG Files|*.jpg"
CD1.FilterIndex = 2
CD1.DefaultExt = "jpg"

CD1.CancelError = True
CD1.ShowOpen
If Trim(CD1.FileName) = "" Then
MsgBox "Canceled By the User"
Exit Sub
Else
fname = CD1.FileName
End If
Img1.Stretch = True
Img1.Picture = LoadPicture(fname)
last:
End Sub
```

Example



Private Sub Command1_Click()

Dim stdinfo As String

Dim stdt As Date

Dim deg(4) As Integer

Dim i As Integer

Dim sum As Integer

Dim average As Single

Text2.Text = ""

stdinfo = "Student Report" + vbCrLf

```
If Text1.Text = "" Then

MsgBox "Student Name required"

Exit Sub

Else

stdinfo = stdinfo + "Student Name: " + Text1.Text + vbCrLf

End If

If Combo1.Text = "" Then

MsgBox "Department Required"

Exit Sub

Else

stdinfo = stdinfo + "Department :" + Combo1.Text + " Engineering"
+ vbCrLf

End If

If List1.ListIndex = -1 Or List2.ListIndex = -1 Then

MsgBox "Please Select Level and/or Section"

Exit Sub

Else

stdinfo = stdinfo + "Level: " + List1.Text + " Section: " + List2.Text
+ vbCrLf

End If

If Option1.Value Then

stdinfo = stdinfo + "Gender: " + "Male" + vbCrLf

End If
```

```
If Option2.Value Then

stdinfo = stdinfo + "Gender: " + "Female" + vbCrLf

End If

stdinfo = stdinfo + Text1.Text + " Can Speak "

If Check1.Value = 1 Then

stdinfo = stdinfo + " Arabic and"

End If

If Check2.Value = 1 Then

stdinfo = stdinfo + " English and"

End If

If Check3.Value = 1 Then

stdinfo = stdinfo + " French"

End If

If Right(stdinfo, 3) = "and" Then

stdinfo = Mid(stdinfo, 1, Len(stdinfo) - 3)

End If

stdinfo = stdinfo + vbCrLf

If Text3.Text = "" Or Not IsDate(Text3.Text) Then

MsgBox "Date of Birth Required"

Exit Sub

Else

stdt = CDate(Text3.Text)

stdinfo = stdinfo + "Date of Birth is: " + Text3.Text + vbCrLf + "
Age is: " + Str((Now() - stdt) \ 365) + " Years" + vbCrLf
```

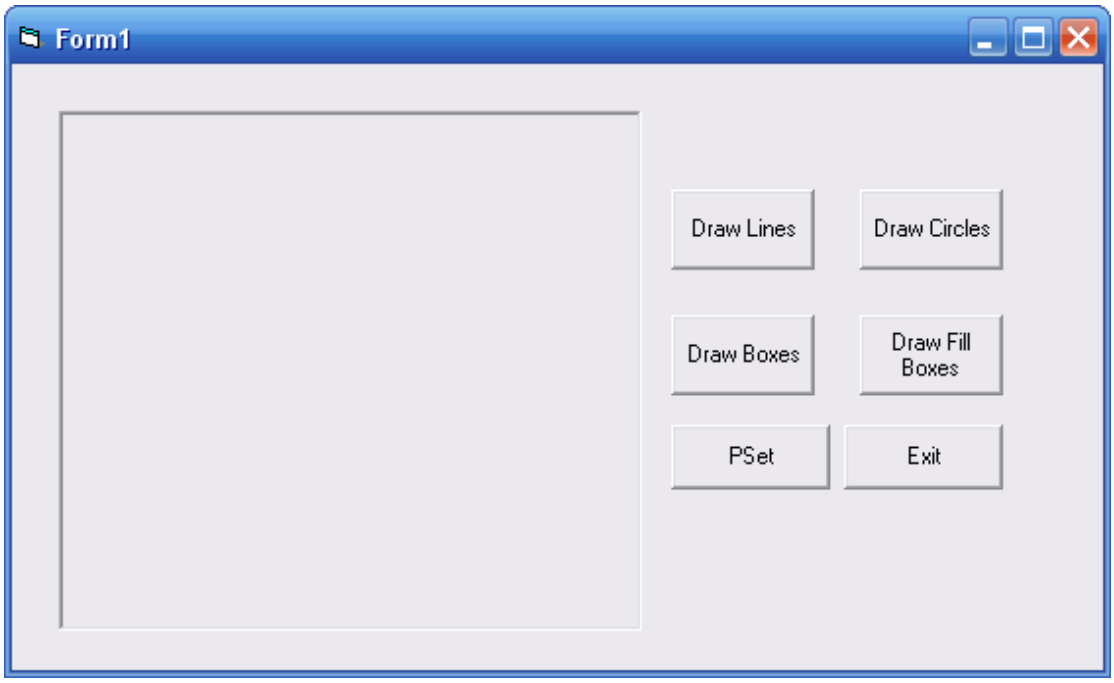
```
End If

If Math.Text = "" Then

Math.Text = InputBox("Enter Degree of Math", "Missing Degree")

End If

deg(1) = Val(Math.Text)

If Survey.Text = "" Then

Survey.Text = InputBox("Enter Degree of Survey", "Missing Degree")

End If

deg(2) = Val(Survey.Text)

If Programming.Text = "" Then

Programming.Text = InputBox("Enter Degree of Programming", "Missing Degree")

End If

deg(3) = Val(Programming.Text)

If Concrete.Text = "" Then

Concrete.Text = InputBox("Enter Degree of Concrete", "Missing Degree")

End If

deg(4) = Val(Concrete.Text)

sum = 0

For i = 1 To 4

sum = sum + deg(i)

Next i

average = sum / 4
```

```
stdinfo = stdinfo + "Degrees for student " + Text1.Text + vbCrLf

stdinfo = stdinfo + "Math = " + Math.Text + vbCrLf

stdinfo = stdinfo + "Survey = " + Survey.Text + vbCrLf

stdinfo = stdinfo + "Programming = " + Programming.Text + vbCrLf

stdinfo = stdinfo + "Concrete = " + Concrete.Text + vbCrLf

stdinfo = stdinfo + "Average = " + Str(average) + vbCrLf

stdinfo = stdinfo + "Student status :"

Select Case Int(average)

Case 90 To 100: stdinfo = stdinfo + "Execlent"

Case 80 To 89: stdinfo = stdinfo + "V. Good"

Case 70 To 79: stdinfo = stdinfo + "Good"

Case 60 To 69: stdinfo = stdinfo + "Intermediate"

Case 50 To 59: stdinfo = stdinfo + "Fair"

Case Is < 50: stdinfo = stdinfo + "Fail"

End Select

stdinfo = stdinfo + vbCrLf

CD1.Flags = cdlCFBoth

CD1.ShowFont

Text2.FontSize = CD1.FontSize

Text2.FontBold = CD1.FontBold

Text2.FontItalic = CD1.FontItalic

Text2.Font = CD1.FontName
```

```
Text2.Text = stdinfo
End Sub

Private Sub Command2_Click()
End
End Sub

Private Sub Command3_Click()
Dim fnum As Integer
Dim fname As String
If Trim(Text2.Text) = "" Then
MsgBox "No Report"
Exit Sub
End If
fname = ""
CD1.DefaultExt = "txt"
CD1.Filter = "Text Files|*.txt"
CD1.ShowSave
fname = CD1.FileName
If Trim(fname) = "" Then
Exit Sub
End If
```

```
fnum = FreeFile()
Open fname For Output As #fnum
Print #fnum, Text2.Text
Close #fnum
End Sub

Private Sub Command4_Click()
CD1.ShowPrinter
Printer.Print Text2.Text
Printer.EndDoc
End Sub
```

Picture box




```
Private Sub Command1_Click()
Dim x, y As Single
P1.Cls
x = P1.Width
y = P1.Height
P1.Line (0, 0)-(x, y), vbRed
P1.Line (0, y)-(x, 0), vbGreen

End Sub

Private Sub Command2_Click()
Dim x, y, z, r As Single
Dim i As Integer
P1.Cls
x = P1.Width
y = P1.Height
If x > y Then
z = y - 10
Else
z = x - 10
End If
r = z / 2
For i = 1 To 4
P1.Circle (z / 2, z / 2), r, vbBlue
```

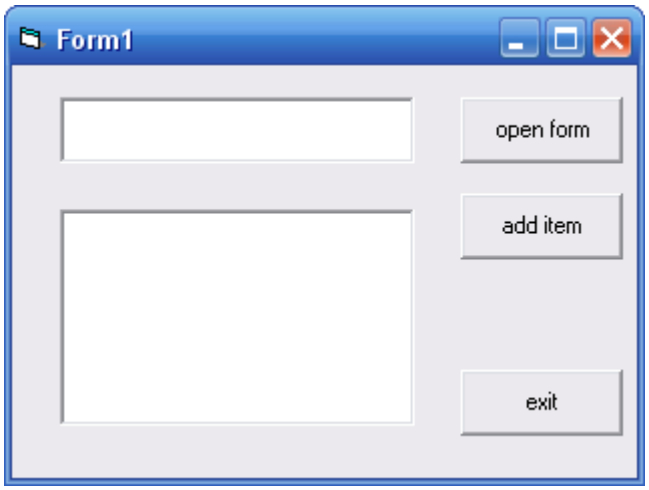
```
r = r / 2
Next i

End Sub

Private Sub Command3_Click()
P1.Cls
x = P1.Width
y = P1.Height
If x > y Then
z = y - 10
Else
z = x - 10
End If
P1.Line (10, 10)-(x / 2, y / 2), vbRed, B
End Sub

Private Sub Command4_Click()
P1.Cls
x = P1.Width
y = P1.Height
If x > y Then
z = y
Else
z = x
End If
P1.Line (x / 2, y / 2)-(x, y), vbBlue, BF

End Sub

Private Sub Command5_Click()
Dim pi As Single
pi = 3.141
P1.Cls
x = P1.Width
y = P1.Height
sc = y / 2
For i = 0 To 2 * pi Step 2 * pi / 360
P1.PSet (i / (2 * pi) * x, y / 2 - sc * Sin(i)), vbRed
Next i

End Sub
```

Private Sub Command6_Click()
End
End Sub

Circle **[**Step**]** (***x, y***), ***radius,* [*color, start, end, aspect*]**
Line **[**Step**]** (***x1***, **y*1***) **[**Step**] -** (***x2***, ***y2***), **[*color*]**, **[**B**][**F**]**
PSet **[**Step**]** (***x***, ***y***), **[*color*]**

```
Private Sub Command7_Click()

Dim pi As Single

pi = 3.141

P1.Cls

P1.Scale (0, 1)-(2 * pi, -1)

For i = 0 To 2 * pi Step 2 * pi / 360

P1.PSet (i, Sin(i)), vbRed

Next i

P1.Scale

End Sub
```

Two Forms



Form1 subroutines
Private Sub Command1_Click()
Form2.Show
End Sub

Private Sub Command2_Click()
End
End Sub

Private Sub Command3_Click()
List1.AddItem Text1.Text
End Sub
Form2 subroutines

```
Private Sub Command1_Click()
Text1.Text = Form1.Text1.Text
End Sub

Private Sub Command2_Click()
Text2.Text = Form1.List1.List(Val(Text3.Text))
End Sub

Private Sub Command3_Click()
Dim i As Integer
List1.Clear
For i = 0 To Form1.List1.ListCount-1
List1.AddItem Form1.List1.List(i)
Next i

End Sub

Private Sub Command4_Click()
Form2.Hide
End Sub
```

Built in Functions

Sin ()
Cos ()
Tan ()
Abs ()
Atn ()
Exp ()

Round(*expression* [*,numdecimalplaces*])

Fix () The difference between **Int** and **Fix** is that if *number* is negative, **Int** returns the first negative integer less than or equal to *number*, whereas **Fix** returns the first negative integer greater than or equal to *number*. For example, **Int** converts -8.4 to -9, and **Fix** converts -8.4 to -
Int ()
Log ()Returns a **Double** specifying the natural logarithm of a number. Log*n(x)* = **Log**(*x*) / **Log**(*n*) The following example illustrates a custom **Function** that calculates base-10 logarithms:

```
Static Function Log10(X)
   Log10 = Log(X) / Log(10#)
End Function
```

Rnd () **Rnd**[**(***number***)**]
The optional *number* argument is a Single or any valid numeric expression.
Return Values

If *number* is	**Rnd generates**
Less than zero	The same number every time, using *number* as the seed.
Greater than zero	The next random number in the sequence.
Equal to zero	The most recently generated number.
Not supplied	The next random number in the sequence.

Remarks
The **Rnd** function returns a value less than 1 but greater than or equal to zero.
The value of *number* determines how **Rnd** generates a random number: For any given initial seed, the same number sequence is generated because each successive call to the **Rnd** function uses the previous number as a seed for the next number in the sequence.

Before calling **Rnd,** use the **Randomize** statement without an argument to initialize the random-number generator with a seed based on the system timer.
To produce random integers in a given range, use this formula:
`Int((upperbound - lowerbound + 1) * Rnd + lowerbound)`
Here, *upperbound* is the highest number in the range, and *lowerbound* is the lowest number in the range.
Note To repeat sequences of random numbers, call **Rnd** with a negative argument immediately before using **Randomize** with a numeric argument. Using **Randomize** with the same value for *number* does not repeat the previous sequence.
Sgn ()

If *number* is	**Sgn returns**
Greater than zero	1
Equal to zero	0
Less than zero	-1

Sqr () Returns a **Double** specifying the square root of a number.

Function	**Derived equivalents**
Inverse Sine	Arcsin(X) = Atn(X / Sqr(-X * X + 1))
Inverse Cosine	Arccos(X) = Atn(-X / Sqr(-X * X + 1)) + 2 * Atn(1)
Hyperbolic Sine	HSin(X) = (Exp(X) – Exp(-X)) / 2
Hyperbolic Cosine	HCos(X) = (Exp(X) + Exp(-X)) / 2
Hyperbolic Tangent	HTan(X) = (Exp(X) – Exp(-X)) / (Exp(X) + Exp(-X))

IsDate(*expression***)**
IsNumeric(*expression***)**
IsEmpty(*expression***)** Returns a **Boolean** value indicating whether a variable has been initialized.
IsNull(*expression***)** Returns a **Boolean** value that indicates whether an expression contains no valid data (Null).

Converting functions
CBool(*expression***)**
CByte(*expression***)**

CCur(*expression***)**
CDate(*expression***)**
CDbl(*expression***)**
CDec(*expression***)**
CInt(*expression***)**
CLng(*expression***)**
CSng(*expression***)**
CStr(*expression***)**
CVar(*expression***)**

Function	**Return Type**	**Range for *expression* argument**
Cbool	Boolean	Any valid **string** or numeric expression.
Cbyte	Byte	0 to 255.
Ccur	Currency	-922,337,203,685,477.5808 to 922,337,203,685,477.5807.
Cdate	Date	Any valid date expression.
CDbl	Double	-1.79769313486232E308 to -4.94065645841247E-324 for negative values; 4.94065645841247E-324 to 1.79769313486232E308 for positive values.
Cdec	Decimal	+/-79,228,162,514,264,337,593,543,950,335 for zero-scaled numbers, that is, numbers with no decimal places. For numbers with 28 decimal places, the range is +/-7.9228162514264337593543950335. The smallest possible non-zero number is 0.0000000000000000000000000001.
Cint	Integer	-32,768 to 32,767; fractions are rounded.
CLng	Long	-2,147,483,648 to 2,147,483,647; fractions are rounded.
CSng	Single	-3.402823E38 to -1.401298E-45 for negative values; 1.401298E-45 to 3.402823E38 for positive values.
CStr	String	Returns for CStr depend on the *expression* argument.
Cvar	Variant	Same range as **Double** for numerics. Same range as **String** for non-numerics.

String functions

LTrim(*string*)
RTrim(*string*)
Trim(*string*)
Left(***string***, ***length***)
Right(***string***, ***length***).
Mid(***string***, ***start***[, ***length***])
Len(*string* | *varname*)
LCase(*string*)
UCase(*string*)
Str() **Asc()** **Chr()**
Val()

Date and Time Functions

Now ()
Date ()
Time () Returns a Variant (Date) indicating the current system time
DateDiff(*interval, date1, date2*[*, firstdayofweek*[*, firstweekofyear*]])
The **DateDiff** function syntax has these named arguments:

Part	**Description**
interval	Required. String expression that is the interval of time you use to calculate the difference between ***date1*** and ***date2***.
date1, ***date2***	Required; **Variant** (**Date**). Two dates you want to use in the calculation.
firstdayofweek	Optional. A constant that specifies the first day of the week. If not specified, Sunday is assumed.
firstweekofyear	Optional. A constant that specifies the first week of the year. If not specified, the first week is assumed to be the week in which January 1 occurs.

Settings

The ***interval*** argument has these settings:

Setting	**Description**
yyyy	Year
q	Quarter
m	Month
y	Day of year
d	Day

w	Weekday
ww	Week
h	Hour
n	Minute
s	Second

```
Dim TheDate As Date   ' Declare variables.
Dim Msg
TheDate = InputBox("Enter a date")
Msg = "Days from today: " & DateDiff("d", Now, TheDate)
MsgBox Msg
```

Format(*expression*[, *format*[, *firstdayofweek*[, *firstweekofyear*]]])
The **Format** function syntax has these parts:

Part	**Description**
expression	Required. Any valid expression.
format	Optional. A valid named or user-defined format expression.
firstdayofweek	Optional. A constant that specifies the first day of the week.
firstweekofyear	Optional. A constant that specifies the first week of the year.

Settings
The *firstdayofweek* argument has these settings:

Constant	**Value**	**Description**
vbUseSystem	0	Use NLS API setting.
VbSunday	1	Sunday (default)
vbMonday	2	Monday
vbTuesday	3	Tuesday
vbWednesday	4	Wednesday
vbThursday	5	Thursday
vbFriday	6	Friday
vbSaturday	7	Saturday

The *firstweekofyear* argument has these settings:

Constant	Value	Description
vbUseSystem	0	Use NLS API setting.
vbFirstJan1	1	Start with week in which January 1 occurs (default).
vbFirstFourDays	2	Start with the first week that has at least four days in the year.
vbFirstFullWeek	3	Start with the first full week of the year.

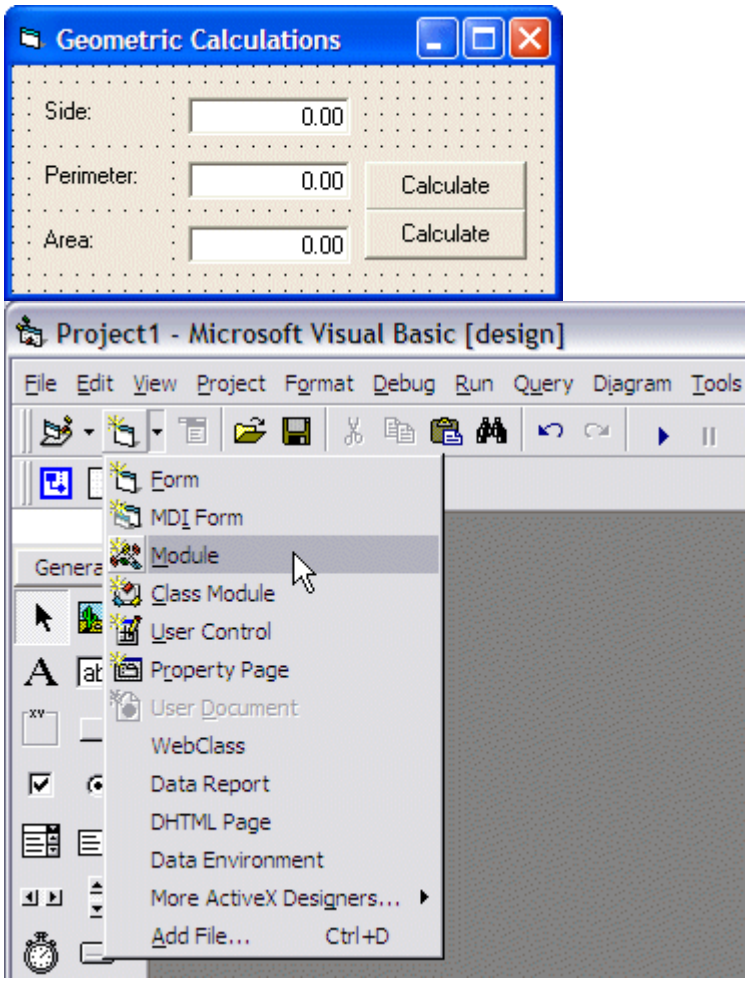
Remarks

To Format	Do This
Numbers	Use predefined named numeric formats or create user-defined numeric formats.
Dates and times	Use predefined named date/time formats or create user-defined date/time formats.
Date and time serial numbers	Use date and time formats or numeric formats.
Strings	Create your own user-defined string formats.

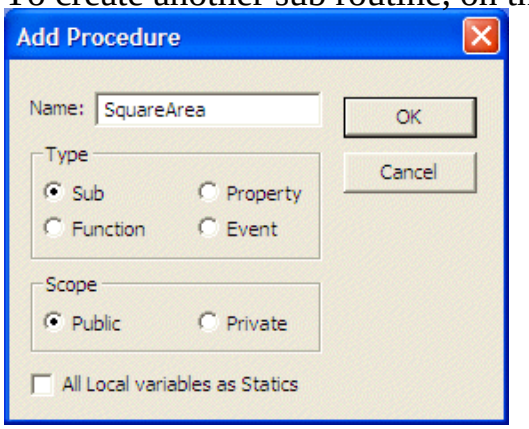
```
Dim MyTime, MyDate, MyStr
MyTime = #17:04:23#
MyDate = #January 27, 1993#

' Returns current system time in the system-defined long
time format.
MyStr = Format(Time, "Long Time")

' Returns current system date in the system-defined long
date format.
MyStr = Format(Date, "Long Date")

MyStr = Format(MyTime, "h:m:s")   ' Returns "17:4:23".
MyStr = Format(MyTime, "hh:mm:ss AMPM")   ' Returns
"05:04:23 PM".
MyStr = Format(MyDate, "dddd, mmm d yyyy")   ' Returns
"Wednesday,
   ' Jan 27 1993".
' If format is not supplied, a string is returned.
MyStr = Format(23)   ' Returns "23".

' User-defined formats.
```

```
MyStr = Format(5459.4, "##,##0.00")   ' Returns
"5,459.40".
MyStr = Format(334.9, "###0.00")   ' Returns "334.90".
MyStr = Format(5, "0.00%")   ' Returns "500.00%".
MyStr = Format("HELLO", "<")   ' Returns "hello".
MyStr = Format("This is it", ">")   ' Returns "THIS IS
IT".
```

Modules, Functions and Procedures



To create another sub routine, on the main menu, click Tools -> Add



Procedure...

```
Sub SquarePerimeter()
    Dim dblSide As Double
    Dim dblPerimeter As Double
```

```
    dblSide = Form1.txtSide
    dblPerimeter = dblSide * 4

    Form1.txtPerimeter.Text = dblPerimeter
End Sub

Public Sub SquareArea()
    Dim dblSide As Double
    Dim dblArea As Double

    dblSide = Form1.txtSide
    dblArea = dblSide * dblSide

    Form1.txtArea.Text = dblArea
End Sub
```

Calling a Sub Routine

```
Private Sub cmdCalcArea_Click()
    SquareArea
End Sub

Private Sub cmdCalcPerimeter_Click()
    SquarePerimeter
End Sub
```

Functions

Introduction

Like a sub routine, a function is used to perform an assignment. The main difference between a sub routine and a function is that, after carrying its assignment, a function gives back a result. We also say that a function "returns a value". To distinguish both, there is a different syntax you use for a function.

Function *FunctionName*() **As** *DataType*

End Function

```
Function RectPerimeter()
    Dim dblLength As Double
    Dim dblHeight As Double
```

```
    dblLength = Form1.txtLength.Text
    dblHeight = Form1.txtHeight.Text

    RectPerimeter = (dblLength + dblHeight) * 2
End Function

Public Function RectArea()
    Dim dblLength As Double
    Dim dblHeight As Double

    dblLength = Form1.txtLength.Text
    dblHeight = Form1.txtHeight.Text

    RectArea = dblLength * dblHeight
End Function
```

Arguments and Parameters

Passing Arguments (By Value)

```
Private Sub txtResult_GotFocus()
    Dim dblHours As Double
    Dim dblSalary As Double

    dblHours = txtHours
    dblSalary = txtSalary

    CalcAndShowSalary dblHours, dblSalary
End Sub

Sub CalcAndShowSalary(Hours As Double, Salary As Double)
    Dim dblResult As Double

    dblResult = Hours * Salary

    txtResult = dblResult

End Sub
```

Passing Arguments By Reference

```
Private Sub Form_Load()
```

```
    Dim FName As String
    FName = "Albert Edou Nkoulou"
    GetFullName FName
    Caption = FName
End Sub

Sub GetFullName(ByRef FullName As String)
    FullName = "Nguyen, Hermine"
End Sub

Function RectPerimeter(ByVal dblLength As Double, _
                       ByVal dblHeight As Double)
    RectPerimeter = (dblLength + dblHeight) * 2
End Function

Sub RectArea(ByVal dblLength As Double, _
             ByVal dblHeight As Double, _
             ByRef dblSurface As Double)
    dblSurface = dblLength * dblHeight
```

End Sub

Built-In Functions

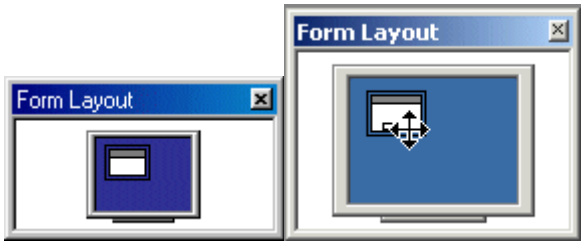
Conversion Functions

Function		
Name	**Return Type**	**Description**
CBool	Boolean	Converts an expression into a Boolean value
CByte	Byte	Converts an expression into Byte number
CDate	Date	Converts and expression into a date or time value
CDbl	Double	Converts an expression into a flowing-point (decimal) number
CInt	Integer	Converts an expression into an integer (natural) number
CCur	Currency	Converts an expression into a currency (monetary) value
CLng	Long	Converts an expression into a long

		integer (a large natural) number
CSng	Single	Converts an expression into a flowing-point (decimal) number
CStr	String	Converts an expression into a string

String-Based Functions

Date and Time Functions
The Date() and Time() functions can be combined and are represented by a function called Now.
Day - Month – Year
Forms
The Startup Position of a Form



Frames

A frame is used as a place holder for other controls; therefore, it is called a container.

Menu

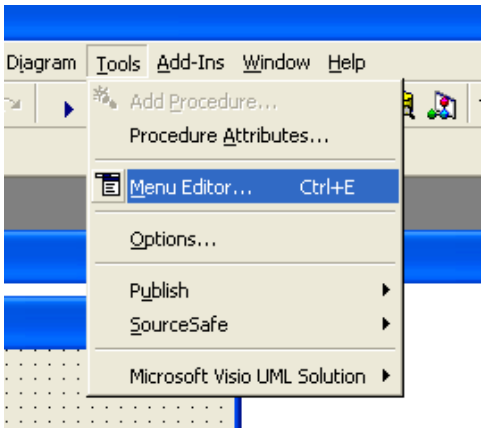
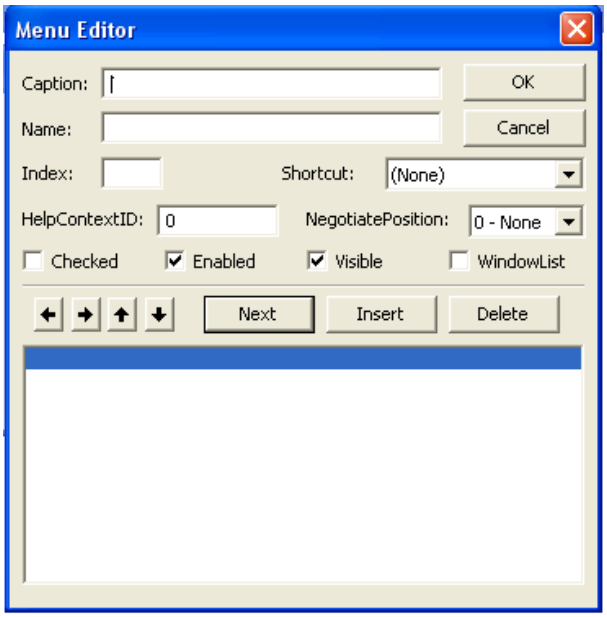
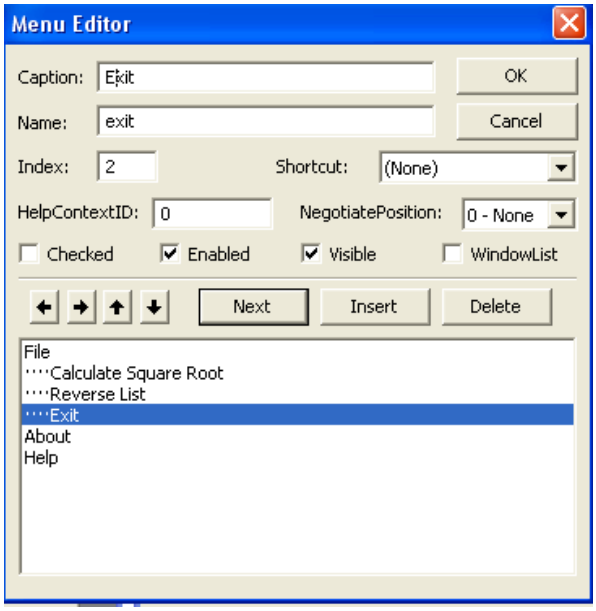
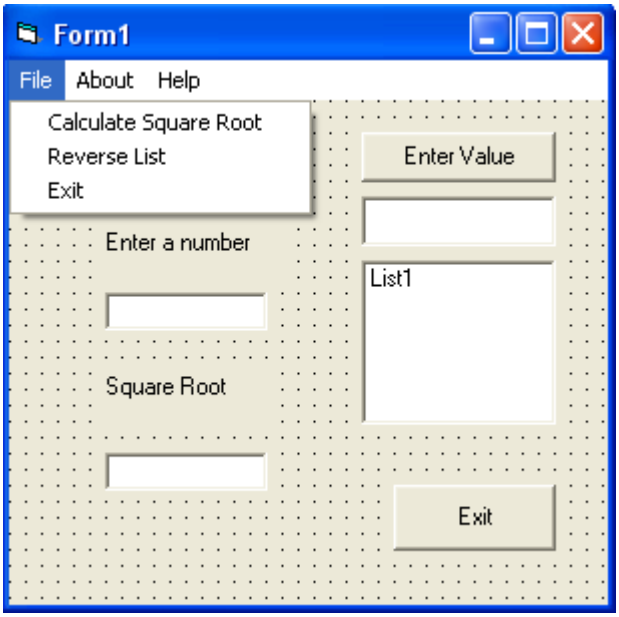


Diagram
Tools
Add-Ins
Window
Help
Add Procedure...
Procedure Attributes...
Menu Editor... Ctrl+E
Options...
Publish
SourceSafe
Microsoft Visio UML Solution


Menu Editor
Caption:
OK
Name:
Cancel
Index:
Shortcut:
(None)
HelpContextID:
0
NegotiatePosition:
0 - None
Checked
Enabled
Visible
WindowList
Next
Insert
Delete


Menu Editor
Caption:
Exit
OK
Name:
exit
Cancel
Index:
2
Shortcut:
(None)
HelpContextID:
0
NegotiatePosition:
0 - None
Checked
Enabled
Visible
WindowList
Next
Insert
Delete
File
····Calculate Square Root
····Reverse List
····Exit
About
Help




```
Private Sub about_Click()
MsgBox "This is a menu driven program"
End Sub

Private Sub calc_Click(Index As Integer)
Dim x As Double
Dim r1 As Double
Dim r2 As Double
If Trim(Text1.Text) = "" Then
MsgBox "Empty text", vbExclamation
Exit Sub
ElseIf Not IsNumeric(Text1.Text) Then
MsgBox "text written is not a number"
Exit Sub
End If
x = Val(Text1.Text)
```

```
r1 = x / 2
                                        ‘r2 = r1
one:
r2 = (x / r1 + r2) / 2                  ‘Do
                                        ‘r1 = r2
If Abs (r1 - r2) >= 0.000001 Then       ‘r2 = (x / r1 + r2) / 2
r1 = r2                                 Loop Until Abs (r1 - r2) < 0.000001
GoTo one
End If
Text2.Text = r2
Text4.Text = Sqr(Val(Text1.Text))
End Sub
Private Sub Command1_Click()
End
End Sub

Private Sub Command2_Click()
If Trim(Text3.Text) = "" Then
MsgBox "Text is Empty", vbInformation
Exit Sub
End If
List1.AddItem Text3.Text
End Sub

Private Sub exit_Click(Index As Integer)
Call Command1_Click
End Sub
```

```
Private Sub help_Click()
MsgBox "Sorry there is no help"
End Sub

Private Sub rev_Click()
 List1 = revlist(List1)
End Sub

Private Function revlist(tmplist As ListBox) As ListBox
Dim x As Integer
Dim n As Integer
Dim s As String
n = tmplist.ListCount
x = n \ 2
For i = 1 To x
s = tmplist.List(i - 1)
tmplist.List(i - 1) = tmplist.List(n - i)
tmplist.List(n - i) = s
Next i
Set revlist = tmplist
End Function
```

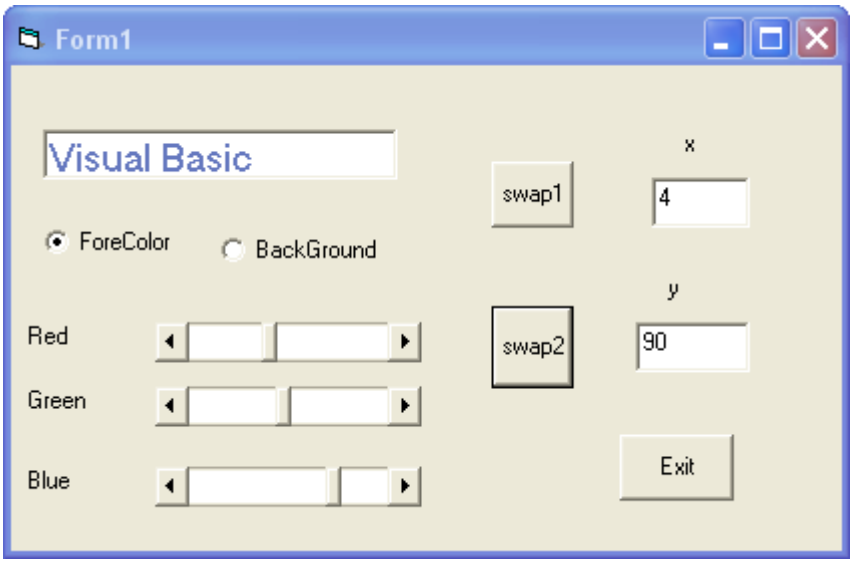
ByRef ByVal

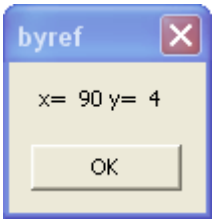
Scrollbar

Small change = 5

Largechange=20

Min=0, max=255



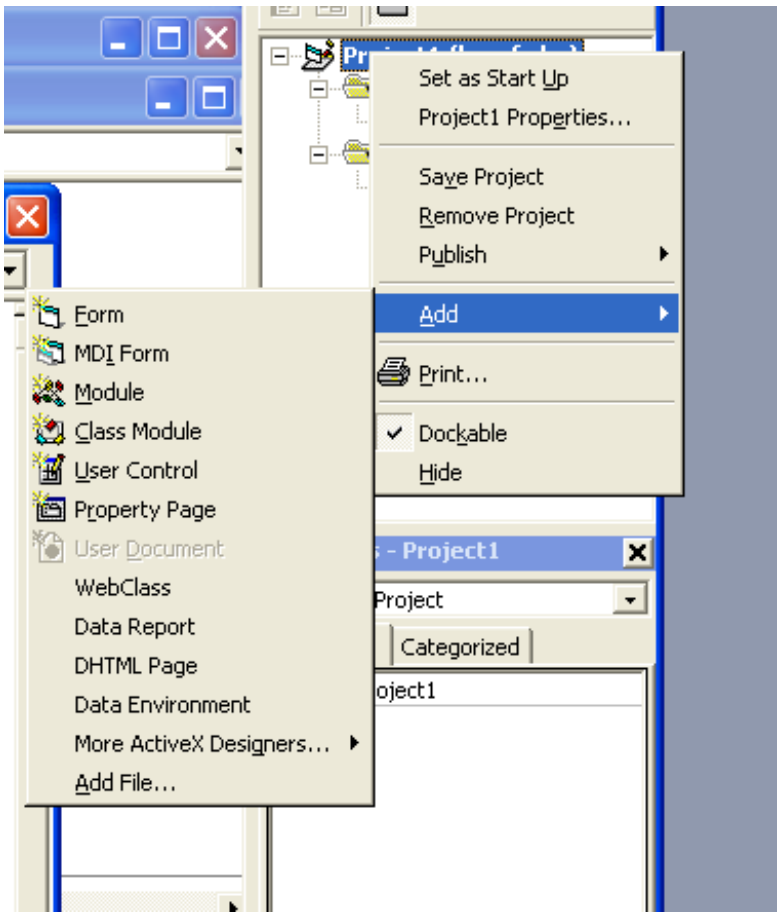


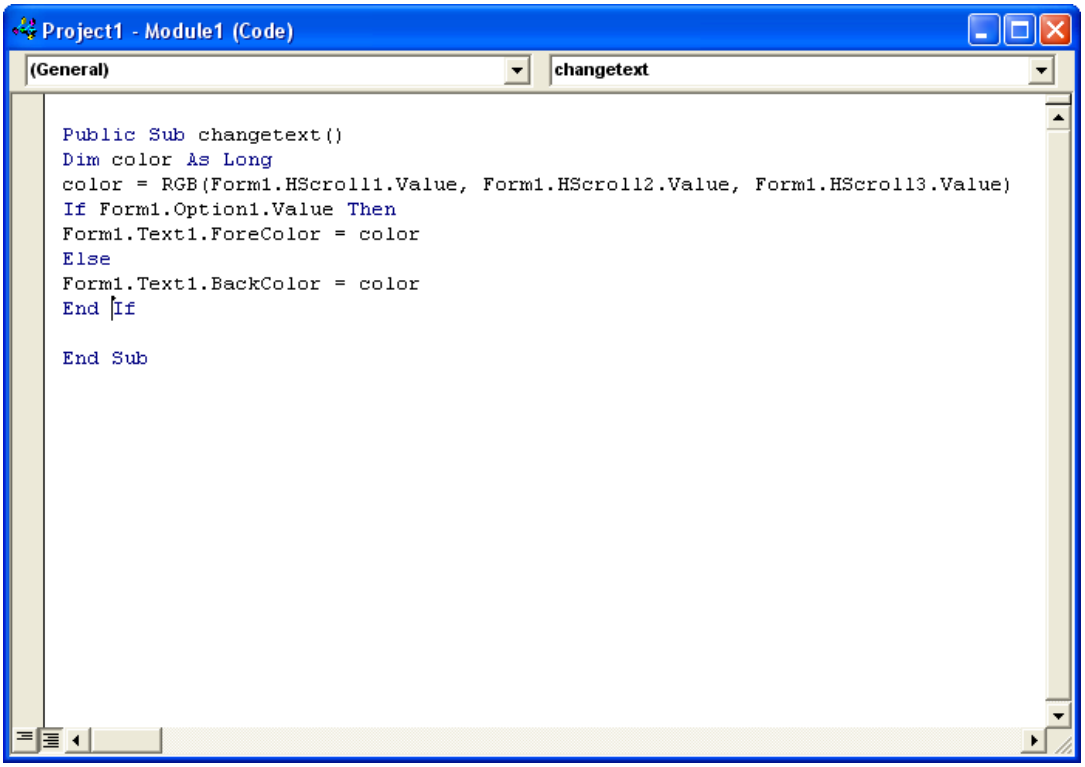

```
Private Sub Command1_Click()
End
End Sub

Private Sub Command2_Click()
```

```
Dim x As Double
Dim y As Double
x = CDbl(Text2.Text)
y = CDbl(Text3.Text)
Call swap1(x, y)
MsgBox "x= " + Str(x) + " y= " + Str(y)
End Sub

Private Sub Command3_Click()
Dim x As Double
Dim y As Double
x = CDbl(Text2.Text)
y = CDbl(Text3.Text)
Call swap2(x, y)
MsgBox "x= " + Str(x) + " y= " + Str(y)
End Sub

Private Sub HScroll1_Change()
Dim color As Long
color = RGB(HScroll1.Value, HScroll2.Value, HScroll3.Value)
If Option1.Value Then
Text1.ForeColor = color
Else
Text1.BackColor = color
End If
End Sub

Private Sub HScroll2_Change()
Dim color As Long
```

```
color = RGB(HScroll1.Value, HScroll2.Value, HScroll3.Value)
If Option1.Value Then
Text1.ForeColor = color
Else
Text1.BackColor = color
End If
End Sub

Private Sub HScroll3_Change()
Dim color As Long
color = RGB(HScroll1.Value, HScroll2.Value, HScroll3.Value)
If Option1.Value Then
Text1.ForeColor = color
Else
Text1.BackColor = color
End If
End Sub

Private Sub swap1(ByVal a As Double, ByVal b As Double)
Dim t As Double
t = a
a = b
b = t
End Sub

Private Sub swap2(ByRef a As Double, ByRef b As Double)
Dim t As Double
t = a
a = b
```

```
b = t
End Sub

Private Sub Text1_DblClick()
CD1.ShowColor
If Option1.Value Then
Text1.ForeColor = CD1.color
Else
Text1.BackColor = CD1.color
End If

End Sub
```




Project1 - Module1 (Code)
(General)
changetext
Public Sub changetext()
Dim color As Long
color = RGB(Form1.HScroll1.Value, Form1.HScroll2.Value, Form1.HScroll3.Value)
If Form1.Option1.Value Then
Form1.Text1.ForeColor = color
Else
Form1.Text1.BackColor = color
End If
End Sub